IMPLEMENTACIÓN DE PROTOCOLO DE ENRUTAMIENTO MULTICAMINO BASADO EN TOPOLOGÍA DE ÁRBOL PARA APLICACIONES DE RECOLECCIÓN DE INFORMACIÓN EN REDES INALÁMBRICAS DE SENSORES

Presentado en cumplimiento de los requisitos para optar al título de

MAGISTER EN INGENIERÍA ELECTRÓNICA

Departamento de Ingeniería Eléctrica y Electrónica

Presentado por

DAINER JULIO VASQUEZ MONROY
Ingeniero Electrónico

Directores de Tesis

JUAN CARLOS VELEZ DÍAZ, Ph.D.
LUIS ALBERTO TORRES HERRERA, M.Sc.

Barranquilla, Colombia

Noviembre 2017

Departamento de Ingeniería Eléctrica y Electrónica

IMPLEMENTACIÓN DE PROTOCOLO DE ENRUTAMIENTO MULTICAMINO BASADO EN TOPOLOGÍA DE ÁRBOL PARA APLICACIONES DE RECOLECCIÓN DE INFORMACIÓN EN REDES INALÁMBRICAS DE SENSORES

Tesis presentada a la *UNIVERSIDAD DEL NORTE* en cumplimiento parcial de los requisitos para optar al título de Magíster en Ingeniería Electrónica

Por

_____

Ing. Dainer Julio Vasquez Monroy

Director de Tesis

_____

Juan Carlos Vélez Díaz, Ph.D.

Director de Tesis

_____

Luis Alberto Torres Herrera, M.Sc.

TABLA DE CONTENIDO

1 INTRODUCCIÓN, FUNDAMENTO TEÓRICO Y OBJETIVOS 5
1.1 INTRODUCCIÓN 5
1.2 FUNDAMENTO TEÓRICO 7
1.3 ANTECEDENTES Y PLANTEAMIENTO DEL PROBLEMA 10
1.4 OBJETIVOS 12
1.4.1 OBJETIVO GENERAL 12
1.4.2 OBJETIVOS ESPECÍFICOS 12
1.5 DELIMITACION DEL PROYECTO 12
1.5.1 ALCANCES 12
1.5.2 LIMITACIONES 13
1.6 CONTRIBUCIONES 13
2 ESTADO DEL ARTE 14
2.1 ENRUTAMIENTO BASADO EN TOPOLOGÍA DE ÁRBOL 14
2.2 TOLERANCIA A FALLAS 17
2.3 ENRUTAMIENTO MULTICAMINO 19
3 IMPLEMENTACIÓN DE PROTOCOLOS DE ENRUTAMIENTO 22
3.1 ENRUTAMIENTO BASADO EN TOPOLOGÍA DE ÁRBOL 22
3.2 PROTOCOLO BASADO EN ÁRBOL DE LA RUTA MÁS CORTA (SPT) 25
3.3 DESCRIPCIÓN DE PROTOCOLO DE ENRUTAMIENTO DE ÁRBOL DOBLE .. 28
3.3.1 Métricas usadas en el protocolo de árbol doble 31
3.3.2 Definición de distancia en el protocolo de árbol doble 32
3.3.3 Ordenando los padres 36
3.3.4 Escalabilidad del protocolo de árbol doble 38
3.3.5 Algoritmo del protocolo de árbol doble 39
3.4 COMPARACIÓN ENTRE EL ÁRBOL SIMPLE Y EL ÁRBOL DOBLE 40
4 RESULTADOS 49
4.1 DEMOSTRACIÓN DE TEOREMA 49
4.2 SIMULACIONES 51
4.2.1 Evaluación de resiliencia 51
4.2.2 Evaluación de retardo 55

5 CONCLUSIONES Y TRABAJO FUTURO 59
5.1 CONCLUSIONES 59
5.2 TRABAJO FUTURO 60

ÍNDICE DE FIGURAS

Figura 1. Diagrama básico de una red inalámbrica de sensores. 9
Figura 2. Red organizada en una topología de árbol simple. 27
Figura 3. Red organizada en una topología de árbol doble. 31
Figura 4. Red con todos los enlaces disponibles. 41
Figura 5. Árbol simple vs Árbol doble, paso 1. 41
Figura 6. Árbol simple vs Árbol doble, paso 2. 42
Figura 7. Árbol simple vs Árbol doble, paso 3. 43
Figura 8. Árbol simple vs Árbol doble, paso 4. 44
Figura 9. Árbol simple vs Árbol doble, paso 5. 45
Figura 10. Árbol simple vs Árbol doble, paso 6. 46
Figura 11. Árbol simple vs Árbol doble, paso 7. 46
Figura 12. Árbol simple vs Árbol doble, paso 8. 47
Figura 13. Árbol simple vs Árbol doble cuando falla el nodo 2. 48
Figura 14. Árbol simple vs Árbol doble cuando falla el nodo 5. 48
Figura 15. Partición del gráfico dirigido inducido por la ruta multicamino con un nodo donde la ruta multicamino colapsa. 49
Figura 16. Comparación del protocolo SPT con el protocolo de árbol doble. 54
Figura 17. Comparación del protocolo SPT con el protocolo de árbol doble. 55
Figura 18. Árbol simple vs Árbol doble 56

1 INTRODUCCIÓN, FUNDAMENTO TEÓRICO Y OBJETIVOS

1.1 INTRODUCCIÓN

Una red inalámbrica de sensores (WSN) consta de muchos dispositivos llamados nodos que miden una variable física del entorno y se comunican inalámbricamente entre sí para enviar sus medidas a un centro de recolección de información [1], [2]. Las redes WSN han sido catalogadas como una de las tecnologías que más desarrollo tendrán en el siglo XXI, ya que pueden tener múltiples aplicaciones dadas sus grandes ventajas sobre otro tipo de sistemas. Sin embargo, estas redes tienen limitaciones importantes en términos de energía, capacidad de procesamiento, radio de transmisión, entre otras [3], [4]. Dichas limitaciones no existen de una manera tan marcada en las redes tradicionales, por tanto, cada aspecto del diseño e implementación que concierne a las WSN debe tener en cuenta esas limitaciones.

Una de las limitaciones más importantes, es la probabilidad relativamente alta de que los nodos de la red fallen, ya que las WSN suelen emplearse para operar en entornos hostiles y pueden agotar su energía rápidamente sin que exista la posibilidad de recargar las baterías. Esos fallos continuos de nodos generan cambios permanentes en la topología de la red, lo cual afecta la cobertura de la red, la confiabilidad en las transmisiones y la conectividad. Por tal motivo, las WSN deben ser resilientes, donde la resiliencia se define como la capacidad de la red para mantener un nivel aceptable en la calidad del servicio al enfrentarse a fallos y desafíos como los descritos anteriormente [5], [6].

Generalmente, las WSN tienen nodos con radio de trasmisión pequeño, de manera que la comunicación entre ellos debe ser mutisalto. Lo anterior implica que la comunicación entre dos nodos puede ser interrumpida si falla un nodo que

pertenece a la ruta entre ellos. En vista de esto, se puede decir que la resiliencia de una WSN depende en gran medida del protocolo de enrutamiento que se emplee para transmitir los mensajes entre los nodos [7]. Existen diferentes clases de protocolos de enrutamiento cuya finalidad es garantizar la comunicación entre dos nodos de una WSN [8], [9], [10]. Sin embargo, una de las principales estrategias empleadas para el diseño de protocolos de enrutamiento que garanticen una resiliencia elevada en caso de fallas de nodos, es el enrutamiento multicamino. El enrutamiento multicamino consiste en descubrir varias rutas disjuntas entre dos nodos en lugar de una sola, de manera que, si la comunicación entre dos nodos se ve interrumpida debido a la falla de un nodo perteneciente a la ruta, pueda hacerse uso de una ruta alternativa [11].

Por otra parte, la estructura de árbol es una de las más empleadas en aplicaciones de recolección de datos [12]. Existen diferentes propuestas con relación a cómo construir un árbol en una WSN. Sin embargo, el tipo de árbol más sencillo en el que cada nodo tiene un padre a través del cual enviar información a la raíz, no es muy resiliente dado que, si un nodo falla, todos sus nodos hijos pueden quedar desconectados de la red. Por tal motivo, existen protocolos de enrutamiento que además de construir un árbol que garantice la comunicación de muchos a uno, emplean enrutamiento multicamino para garantizar la resiliencia de la red [13], [14].

En este contexto, en el presente informe se describe la implementación de un protocolo de enrutamiento multicamino basado en topología de árbol cuya finalidad es favorecer la conectividad de los nodos y la confiabilidad en la transmisión de los paquetes a medida que van muriendo nodos en una WSN. El protocolo de enrutamiento implementado es llamado *árbol doble,* y ofrece la siguiente garantía: si un nodo falla, todos los nodos restantes continúan teniendo al menos una ruta hasta la raíz, sin importar qué nodo ha fallado. Los resultados obtenidos demuestran que la implementación realizada favorece la conectividad y ofrece más garantías de resiliencia con respecto al algoritmo *Shortest path tree (SPT).* sin

incrementar el retardo en los paquetes enviados por los nodos por encima del 1%. El *árbol doble* solo necesita que los nodos empleen información local de la red, a diferencia de los protocolos H-SPREAD y DACA, presentados en [13] y [14] respectivamente. La implementación se realizó en el simulador OMNET++ versión 4.6 haciendo uso de algunos módulos de MiXim versión 2.3.

1.2 FUNDAMENTO TEÓRICO

Una red inalámbrica de sensores (WSN) consiste en un conjunto de nodos distribuidos en un área geográfica que miden una variable física del entorno, procesan la información obtenida y colaboran entre sí para enviarla inalámbricamente a una estación remota [1], [2]. La principal ventaja de las WSN radica en que a pesar de las complejas tareas que realizan, sus nodos son de tamaño reducido y pueden operar en entornos hostiles, lo cual hace que tengan múltiples aplicaciones en sistemas de monitoreo, vigilancia, automatización de procesos, asistencia en casos de desastres, microcirugías, agricultura, la industria militar, entre otros [3]. Sin embargo, las WSN presentan características que suponen grandes desafíos para su funcionamiento, entre las cuales se cuentan múltiples limitaciones en términos de energía, capacidad de procesamiento y almacenamiento también limitada, radio de transmisión pequeño, presencia de cientos o miles de nodos y cambios permanentes en la topología de la red debido a fallas o adición de nodos [4].

Una de las desventajas más importantes en las WSN, es que los nodos pueden fallar con frecuencia debido a que pueden operan en entornos hostiles y generalmente no se les puede recargar la batería. Además, muchas veces no se puede acceder físicamente a los nodos para programarlos una vez que han sido desplegados en el entorno. Por tal motivo, las WSN deben ser autoconfigurables y tener propiedades de auto curación que garanticen una alta confiabilidad en las transmisiones cuando se presenten fallas de nodos o enlaces, incrementándose así la resiliencia de la red [44].

Hay dos estructuras de topología bastante comunes en las WSN, que son: La topología de árbol y la topología de malla. En la topología de árbol, cada nodo tiene un nodo padre al cual siempre envía los datos de aplicación. Por otra parte, en la topología de malla existen una serie de múltiples conexiones entre los nodos, de manera que un mismo nodo no siempre envía los datos de aplicación al mismo nodo vecino. La ventaja de la topología de árbol es que facilita el enrutamiento, ya que cada nodo solo necesita almacenar la identidad del nodo padre en su tabla de enrutamiento. Sin embargo, la topología de árbol tiene problemas de resiliencia, ya que, si un nodo falla, todos sus hijos corren el riesgo de quedar desconectados de la red. Por otra parte, en la topología de malla deben almacenarse tablas de enrutamiento más complejas que en la topología de árbol, de manera que cada nodo sepa qué ruta usar en una situación particular. Sin embargo, la topología de malla generalmente es más resiliente que la topología de árbol en lo que respecta a la tolerancia a fallos y desafíos [45].

Las topologías descritas anteriormente, generalmente se crean mediante un proceso de intercambio de mensajes de configuración, el cual tiene lugar antes del envío de datos de aplicación. Este hecho les representa una desventaja importante con relación a protocolos que no requieren la construcción previa de las rutas por las cuales se envían los datos de aplicación [38].

Con relación a las aplicaciones más comunes en que se emplean las redes WSN, una de las más importantes consiste en el monitoreo de un entorno [46]. Para efectuar el monitoreo, los nodos deben recolectar información del entorno y enviarla a una estación central o raíz siguiendo las reglas de un protocolo de enrutamiento. La figura 1 presenta el diagrama básico de una red de sensores en aplicaciones de este tipo.

Como se observa en la figura 1, una red WSN puede contener cientos o miles de nodos distribuidos aleatoriamente en un espacio geográfico que puede ser de

difícil acceso y hostil. En aplicaciones de recolección de información, los nodos deben enviar mensajes a un nodo llamado raíz, el cual se comunica con un centro de recolección que posee más capacidad de procesamiento y almacenamiento que los nodos de la red [22]. Debido a estas características, el centro de recolección puede procesar y transmitir la información recibida a una red mayor, de manera que un usuario ubicado remotamente pueda visualizar la información obtenida por los nodos. Las redes de recolección de información pueden tener varias raíces, dependiendo de la aplicación particular [15].

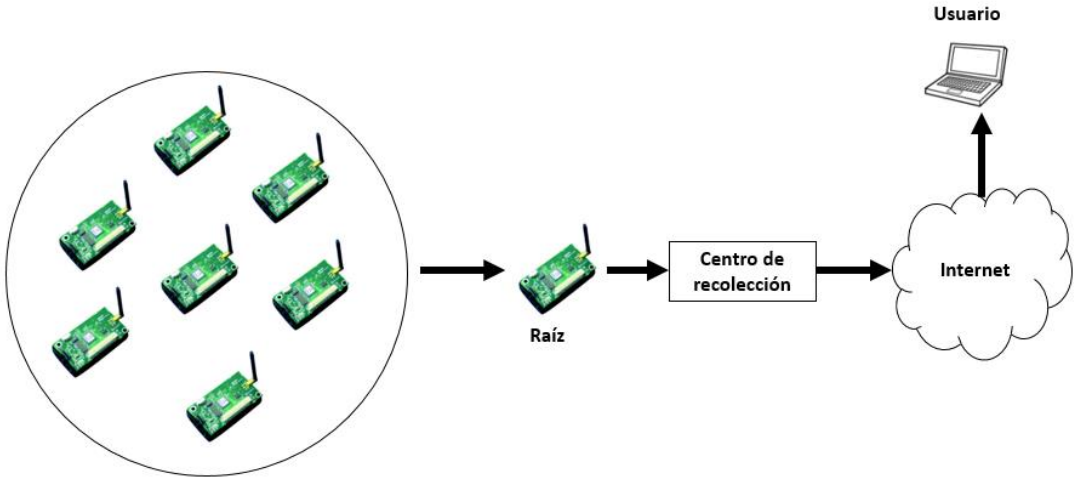

Figura 1. Diagrama básico de una red inalámbrica de sensores.

Las aplicaciones de recolección de información como la que se ilustra en la figura 1, generalmente siguen un patrón de comunicación de muchos a uno, es decir, todos los nodos de la red deben enviar información a la raíz. Dadas las limitaciones de potencia de transmisión de los nodos de una red WSN, la comunicación debe ser mutisalto, de manera que los nodos no solo envían información propia, sino que también retransmiten información proveniente de otros nodos.

1.3 ANTECEDENTES Y PLANTEAMIENTO DEL PROBLEMA

Actualmente, existen estrategias orientadas a garantizar la confiabilidad y la tolerancia a fallos y desafíos en una WSN que sigue un patrón de comunicación de muchos a uno. En este contexto, en [14] se describe la implementación del protocolo DACA, el cual consta principalmente de tres fases. La primera fase consiste en seleccionar un sub-conjunto de nodos que se mantendrán operativos mientras el resto de los nodos duerme. El objetivo de esta fase es mantener la mayor cobertura posible del entorno con la menor cantidad posible de nodos encendidos, ahorrando así energía y prolongando el tiempo de vida de la red. La segunda fase propone un mecanismo para prender algunos nodos dormidos en caso de que uno de los nodos activos falle, de manera que la red continúe garantizando un adecuado nivel de resiliencia y cobertura. La tercera fase consiste en encontrar múltiples rutas disjuntas desde cada nodo de la red hasta la raíz empleando la estrategia descrita en H-SPREAD [13], la cual consiste en un protocolo de enrutamiento multicamino que tiene dos fases. En la primera fase, se construye un árbol simple donde cada nodo posee una ruta para enviar los datos de aplicación hasta la raíz. El árbol se construye mediante el intercambio de mensajes de configuración. En la segunda fase, se intercambian mensajes de configuración adicionales para que cada nodo encuentre la mayor cantidad posible de rutas disjuntas en nodos hasta la raíz. Así, si un nodo falla, los datos pueden desviarse a una ruta alternativa.

Tanto H-SPREAD [13] como DACA [14] presentan limitaciones importantes, ya que el mecanismo de búsqueda de rutas implica un intercambio de mensajes de configuración adicional a los mensajes que se necesitan para la construcción de un árbol simple de un solo padre. Además, los protocolos mencionados tienen problemas de escalabilidad, ya que esos mensajes de configuración adicionales contienen las rutas completas hasta la raíz, elevando considerablemente el *overhead*. Además, posiblemente no se requieran todas las rutas disjuntas en nodos desde cada nodo de la red hasta la raíz, de manera que podrían disminuirse

el número de mensajes de configuración intercambiados si se flexibiliza tal requerimiento.

En algunos casos, posiblemente solo se requiera tener la garantía de que, si M nodos fallan, los nodos restantes continúan conectados a la WSN, proveyendo así una medida concreta de la resiliencia de la red. Con relación a este punto, se han propuesto algunos algoritmos para construir una red *k-connected*, es decir, una red donde todos los nodos tienen una ruta para comunicarse entre sí aun cuando fallan k-1 nodos en la red [6].

En el presente trabajo, se propone una estrategia llamada *árbol doble,* en la que cada nodo de la red posee dos rutas disjuntas en nodos hasta la raíz. La estrategia propuesta ofrece la siguiente garantía: si un nodo falla, los nodos restantes continúan teniendo al menos una ruta hasta la raíz, sin importar la localización del nodo que ha fallado. Dado que todos los datos de aplicación tienen como destino a la raíz, se podría decir que la anterior garantía asegura que la red es *k-connected* con k=2. Para encontrar las rutas, se emplea la misma cantidad de mensajes de configuración que se necesitan para construir un árbol simple, disminuyendo así el tráfico de control en comparación con [14]. Además, en *árbol doble* los nodos emplean información local de la red, no información global, como sucede en [14].

1.4 OBJETIVOS

1.4.1 OBJETIVO GENERAL

- Implementar y evaluar un protocolo de enrutamiento multicamino destinado a incrementar la tolerancia a fallos y desafíos en redes inalámbricas de sensores (WSN).

1.4.2 OBJETIVOS ESPECÍFICOS

- Diseñar las reglas mediante las cuales operará el protocolo implementado.
- Implementar el protocolo de enrutamiento en una herramienta de simulación.
- Evaluar el protocolo implementado en términos de la tolerancia a fallos progresivos de nodos en una red WSN.

1.5 DELIMITACION DEL PROYECTO

1.5.1 ALCANCES

- Se considerarán redes de sensores de nodos distribuidos uniformemente en espacio de dos dimensiones. Los nodos se asumen homogéneos.
- El protocolo de enrutamiento implementado será una adaptación de uno de los protocolos del estado del arte junto con algunas modificaciones propias.

- Se implementará un segundo protocolo del estado del arte contra el que se pueda realizar una comparación.

1.5.2 LIMITACIONES

- La implementación y evaluación del protocolo se realizará en el simulador OMNET++ 4.6 y empleando la herramienta de simulación MiXim 2.3, de manera que no se hará una implementación real del sistema.

- No se evaluará el protocolo cuando haya envío de datos de aplicación en la red.

1.6 CONTRIBUCIONES

- El protocolo implementado no necesita de intercambio de mensajes de configuración durante la fase de transmisión de datos de aplicación, de manera que el árbol no se calcula continuamente.

- La cantidad de mensajes de configuración que se envían durante la fase de construcción de *árbol doble* es la misma que se emplea para construir un árbol simple de un solo padre.

- La resiliencia de *árbol doble* es superior a la del árbol empleado en el protocolo SPT a medida que fallan nodos en la WSN.

- A diferencia de H-SPREAD [13] y DACA [14], en *árbol doble* los nodos no requieren recibir ni enviar información global de la red para calcular las rutas.

- El protocolo *árbol doble* ofrece la garantía de que la red es k-connected para k=2.

2 ESTADO DEL ARTE

2.1 ENRUTAMIENTO BASADO EN TOPOLOGÍA DE ÁRBOL

Una de las topologías más empleadas para la recolección de datos en una WSN es la de árbol. La topología de árbol que sigue un patrón de comunicación de muchos a uno se denomina Convergecast [16]. Es posible que en la red haya más de un nodo que actúe como raíz. En este caso, se puede construir una topología de árbol distinta por cada raíz, garantizando una ruta desde cada nodo de la red hasta cada raíz [15].

Entre las ventajas de la topología de árbol, se cuentan que garantiza que cada nodo de la red tenga al menos una ruta hasta la raíz en caso de que la conectividad exista. Además, la topología de árbol es distribuida, de manera que cada nodo solo necesita conocer información sobre sus vecinos, no sobre todos los nodos de la red. La naturaleza distribuida de la topología favorece también la escalabilidad, simplificando el proceso de adición de nodos a la red [16]. La principal desventaja de esta topología es que requiere el intercambio de mensajes de configuración para construirse, lo cual debe llevarse a cabo antes del envío de datos de aplicación [17].

Se han propuesto muchos protocolos y algoritmos para construir un árbol en aplicaciones de recolección de datos. La mayoría de estos se concentran en maximizar el tiempo de vida de la red [18] - [20], recolectar los datos con la mayor rapidez posible disminuyendo los retardos en los paquetes [21] y minimizar el costo de energía en árboles que emplean agregación de datos en el envío de datos hasta la raíz [23]. En la tabla 1 se presentan algunos de estos algoritmos.

Uno de los árboles más empleados es el árbol de la ruta más corta (SPT) [24]. En SPT, cada nodo tiene la ruta más corta (de menor distancia) hasta la raíz, donde

la distancia está dada en términos de una métrica como puede ser el número de saltos hasta la raíz. SPT puede ser construido empleando una versión distribuida del algoritmo de Dijkstra. El proceso de construcción del árbol empieza en la raíz, que envía un mensaje de configuración a los nodos vecinos publicando su identidad y distancia igual a 0. El resto de los nodos empieza con distancia infinita. Cada vez que un nodo recibe un mensaje de configuración, el nodo calcula su distancia a la raíz a través de ese remitente. Si la distancia calculada resulta ser inferior a su distancia actual, el nodo actualiza tanto su distancia hasta la raíz como la identidad de su padre. Además de lo anterior, el nodo envía a sus vecinos su propio mensaje de configuración con su distancia actualizada. El proceso termina una vez que todos los nodos han enviado su propio mensaje de configuración y el árbol se ha estabilizado.

Otro de los árboles más empleados es *Breadth-first search (BFS),* el cual también garantiza que cada nodo posee la ruta más corta hasta la raíz [25]. La particularidad de BFS es la forma como se construye el árbol. El algoritmo primero encuentra el número de vecinos que tiene la raíz, los cuales están a un salto de esta. Luego, el algoritmo encuentra todos los nodos que están a dos saltos de la raíz y los conecta a la red. El proceso se repite hasta que los nodos que están a n saltos de la raíz se han conectado a la red, donde n es la distancia en saltos del nodo más distante a la raíz a través de la ruta más corta. En BFS, los nodos que están a k saltos de la raíz solo se conectan a la red cuando todos los nodos de menor distancia en saltos ya están conectados.

Debido a que los sensores en una WSN pueden operan en entornos hostiles, hay una probabilidad elevada de que los nodos fallen. En estos casos, debe garantizarse un elevado nivel de resiliencia a medida que los nodos de la red van fallando, donde resiliencia se define como la capacidad de la red de mantener un nivel aceptable en la calidad del servicio al enfrentarse a fallos de nodos o enlaces [5], [6], [30]. Por esta razón, la gestión de la topología es un aspecto importante en las WSN. Con relación a este punto, un árbol simple donde cada nodo tiene un

solo padre no es muy resiliente, dado que, si un nodo falla, todos sus nodos hijos corren el riesgo de quedar desconectados de la red. Por tanto, se requieren algunos mecanismos para limitar el número de nodos que permanecen desconectados en presencia de fallos de nodos o enlaces.

Un protocolo de enrutamiento basado en topología de árbol que garantiza un adecuado nivel de resiliencia es Collection Tree Protocol (CTP) [26], [32]. En CTP, se construye un árbol dinámico que se actualiza periódicamente mediante el intercambio permanente de mensajes de configuración. CTP ha sido implementado en simuladores para redes de sensores como Castalia [27] o TOSSIM [28]. La principal desventaja de protocolos de este tipo es que incrementan considerablemente el *overhead*, dado que los mensajes de configuración empleados para recalcular continuamente el árbol conviven junto con el envío de datos de aplicación. Sin embargo, este tipo de protocolos proveen altos niveles de resiliencia, ya que el cálculo continuo del árbol hace que los nodos encuentren rutas hasta la raíz durante la fase de operación de la red en presencia de fallas de nodos [29].

Por otra parte, en [17] se propone el protocolo de enrutamiento EDGES, el cual se basa en topología de árbol y no requiere de intercambio permanente de mensajes de configuración. Para sobreponerse a fallas de nodos o enlace, EDGES guarda padres alternativos en la configuración inicial siguiendo unas reglas particulares, de manera que cuando un padre falla, los datos de aplicación sean desviados a un padre alternativo sin necesidad de que el árbol deba ser recalculado. Aunque el protocolo propuesto es escalable, un nodo puede requerir información sobre otro que está a dos saltos de distancia, y no solamente a uno tal como ocurre en la lógica de funcionamiento más simplificada de un protocolo de árbol típico. Además, EDGES no garantiza que, si un nodo falla, sus hijos automáticamente tengan otra ruta a través de la cual enviar mensajes hasta la raíz.

Algoritmo	Variable que minimiza o maximiza	Objetivos	Tipo de red
ADCMCST [47]	Minimizar número de nodos hijos por nodo	Balancear tráfico, Prolongar tiempo de vida	Nodos homogéneos
[48]	Maximizar energía residual en los nodos y minimizar número de saltos (Cuando entran en conflicto, no establece cuál tendrá prioridad)	Prolongar tiempo de vida	Nodos homogéneos
[49]	Minimizar número de saltos entre nodos líderes (La red es dividida en clusters)	Prolongar tiempo de vida	Nodos homogéneos
[50]	Maximizar número de nodos hijos por nodo	Reducir número de nodos huérfanos	Nodos heterogéneos
[24]	Minimizar longitud de la ruta	Reducir retardo	Nodos homogéneos
Árbol doble	Minimizar longitud de la ruta bajo la restricción de que cada nodo debe tener dos padres antes de extender el árbol, excepto la raíz y sus hijos	Aumentar resiliencia	Nodos homogéneos

Tabla 1. Comparación de algoritmos que construyen árbol.

2.2 TOLERANCIA A FALLAS

En una WSN, los nodos y enlaces pueden fallar con frecuencia, de manera que el protocolo de enrutamiento empleado debe garantizar que la red mantendrá altos niveles de confiabilidad en las transmisiones aún si ocurren fallas. Cuando la topología es el árbol simple, la muerte de un nodo implica la desconexión de todos sus hijos, lo cual puede particionar la red o hacerla inoperante a pesar de que haya una gran cantidad de nodos funcionando adecuadamente. Por tal motivo, es necesario que existan técnicas de gestión de topología que permitan mantener la red funcionando en estos casos. De acuerdo con [5] y [31], las técnicas que hay actualmente en el estado del arte con relación a este punto, se enfocan en el descubrimiento de nodos, la gestión de ciclo de sueño, el clustering, el control de potencia de transmisión y el control de movimiento. Este trabajo se concentra en formar una topología inicial que no se actualiza para reducir al mínimo el tráfico de control. Por tal motivo, es de interés el aspecto relacionado con el descubrimiento de nodos, el cual se basa en detectar oportunamente los nodos que hay en la red

no solo en el proceso de configuración inicial, sino también durante su etapa de operación. Cuando se detecta la presencia de más nodos, la densidad se incrementa, lo cual aumenta la posibilidad de encontrar rutas alternativas hacia el destino en caso de fallas.

Por otra parte, cuando una red es tolerante a fallas se dice que es resiliente. Se han establecido algunos mecanismos para medir la resiliencia en una WSN. Uno de los más importantes es el del *k-connected*. Se dice que una red es *k-connected* si dos nodos cualesquiera pueden establecer una comunicación mutisalto aún si fallan k-1 nodos en la red, sin importar qué nodos fallen [6].

En el presente trabajo, se desea garantizar cierto nivel de resiliencia de la red, aunque se tienen varias limitaciones. Entre estas limitaciones se cuentan que no se puede controlar la posición de los nodos, no se pueden añadir nodos a la red y tampoco puede ajustarse la potencia de transmisión. Especialmente, se desea garantizar que la red es *k-connected* para k=2, donde el concepto de *k-connected* que se ha empleado varía con relación al concepto típico descrito en [6].

En este trabajo, se dirá que una red es *k-connected* para k=2 si todos los nodos continúan teniendo al menos una ruta hasta la raíz en caso de que un nodo de la red falle, sin importar la localización del nodo que falló. El anterior es un concepto menos exigente que el descrito en [6], sin embargo, es apropiado dado que el patrón de comunicación en las WSN generalmente es de muchos a uno, no de muchos a muchos. A menos que se indique lo contrario, de aquí en adelante se empleará la expresión *k-connected* en el sentido descrito anteriormente, no en el sentido descrito en [6].

De acuerdo con [5], [6] y [31], uno de los aspectos de diseño que más atención requieren para implementar una WSN resiliente, es el enrutamiento. El enrutamiento consiste en el descubrimiento, mantenimiento y utilización de rutas para transmitir mensajes de un nodo a otro en una red. Dado que una red *k-*

connected requiere que se tenga una ruta entre dos nodos aún si fallan k-1 nodos, el enrutamiento debe encargarse de encontrar dichas rutas y establecer un mecanismo que dictamine cómo se emplearán esas rutas a medida que vayan fallando nodos en la red.

2.3 ENRUTAMIENTO MULTICAMINO

Existe una gran cantidad de protocolos de enrutamiento, los cuales pueden ser clasificados de acuerdo con la estructura de la red o al protocolo de operación. Con relación a la estructura de la red, el enrutamiento puede ser plano, Jerárquico o basado en la localización. Por otra parte, de acuerdo con el protocolo de operación, el enrutamiento puede ser basado en negociación, basado en consulta, multicamino, centrado en datos, entre otros [8] - [10]. En este trabajo, es de interés el enrutamiento plano, dado que los nodos son homogéneos.

El tipo de enrutamiento empleado en una WSN depende de la aplicación en la que la red se utilice, las características de la red y los requerimientos en la calidad del servicio. En este trabajo en particular, se desean aprovechar las propiedades deseables del enrutamiento multicamino para incrementar la confiabilidad y la tolerancia de la red a fallas de nodo o enlace, lo cual se relaciona directamente con la resiliencia de la red. De hecho, el enrutamiento multicamino es una de las técnicas más empleadas para tal fin [5]. El enrutamiento multicamino consiste en proveer a cada nodo de la red múltiples rutas distintas hacia el destino en lugar de una única ruta. Esto permite que, si un nodo de la ruta principal falla, se haga uso de una ruta alternativa.

Según [33], dos rutas distintas entre dos nodos pueden ser:

- *Disjuntas en nodos:* Son rutas que no comparten nodos. Son las más resistentes a fallos de nodos.

- *Disjuntas en enlaces:* Son rutas que no comparten enlaces.

- *Parcialmente disjuntas:* Son rutas distintas que comparten algunos nodos y enlaces.

  De lo anterior se deduce que, si existen N rutas disjuntas en nodos entre los nodos A y B, la conexión entre ambos siempre es posible aún si fallan N – 1 nodos en la red, lo cual se relaciona con el concepto de *k-connected* presentado en la sección anterior.

  De acuerdo con [33], El enrutamiento multicamino es empleado para diversos propósitos, entre los cuales se cuentan los siguientes:

- *Incrementar la confiabilidad y la tolerancia a fallos:* Si se tienen varias rutas distintas de un nodo fuente a un nodo destino, es posible que la transmisión de un paquete se produzca aun cuando existan fallas de nodos o enlaces a lo largo de la ruta principal, lo cual sería imposible si hubiese una sola ruta disponible entre los nodos. Lo anterior puede lograrse mediante dos formas diferentes. La primera de ellas es desviando los datos de aplicación a la ruta alternativa en caso de fallas en la ruta principal. La segunda forma es transmitir el paquete por varias rutas, de manera que, si una de las rutas presenta un fallo, el paquete transmitido aún tenga probabilidades de llegar al destino. En el presente trabajo, la técnica que se emplea en el protocolo implementado es la de desviar los datos de aplicación a la ruta alternativa en caso de que falle un nodo perteneciente a la ruta principal [42].

- *Mejorar la calidad del servicio (QoS):* Las rutas disponibles entre dos nodos pueden clasificarse de acuerdo con una métrica que determine la calidad –por ejemplo, el retardo-, de manera que los paquetes más importantes se transmitan por la ruta que ofrezca mejor calidad.

- *Prolongar el tiempo de vida:* Al disponer de varias rutas hacia la raíz, los nodos pueden emplear aquellas cuyos nodos dispongan de mayor energía. De esta forma, la energía puede consumirse de manera uniforme en la red, evitando particiones prematuras que limiten su tiempo de vida [36].

  Otros objetivos que se persiguen al emplear enrutamiento multicamino son: incrementar la seguridad [35] y balancear el tráfico [37].

  Algunos desafíos asociados al diseño de protocolos de enrutamiento multicamino, incluyen el diseño del mecanismo de búsqueda de rutas disjuntas, la minimización del número de mensajes intercambiados en la fase de descubrimiento de rutas disjuntas [38], la selección de las rutas a utilizar, la distribución y el balanceo del tráfico, el mantenimiento de las rutas, entre otros. Además, las rutas disjuntas pueden interferirse por estar demasiado cerca unas de otras en caso de que sean usadas simultáneamente, lo cual podría disminuir la calidad del servicio [39] - [41].

  La gran mayoría de protocolos de enrutamiento multicamino presentados en [33], se concentran en la búsqueda de rutas disjuntas entre dos nodos particulares de la red. Sin embargo, el patrón de comunicación en una WSN generalmente es de muchos a uno, es decir, múltiples nodos que recolectan datos y los envían a una raíz. Una forma elemental y eficiente de implementar una red que sigue este patrón consiste en usar una topología de árbol de una raíz, tal como se explicó anteriormente. Sin embargo, cuando una falla de nodo ocurre, gran parte de la red puede quedar desconectada. Por tal motivo, en [14] se presenta el protocolo DACA, el cual construye un árbol con una raíz mediante el intercambio de mensajes de configuración. Además, el protocolo sugiere intercambiar mensajes de configuración adicionales que permitan encontrar múltiples rutas disjuntas en nodos desde cualquier nodo de la red hasta la raíz. Este proceso de descubrimiento de rutas permite incrementar los niveles de confiabilidad de la red ante fallas de nodos. Sin embargo, el mecanismo tiene algunos problemas de escalabilidad, ya que, de acuerdo con sus reglas de operación, cada nodo debe almacenar todas las rutas disjuntas hasta la raíz e irlas transmitiendo como parte

de sus mensajes de configuración, contrario a lo que ocurre en los protocolos de árbol típicos donde solo se transmite la identidad del nodo que envía el mensaje de configuración y su distancia a la raíz. Además, en los protocolos basados en topología de árbol, solo se almacenan las identidades de los posibles siguientes padres a los cuales se deben enviar los datos de aplicación, no las rutas completas.

En vista de lo anterior, en el presente trabajo se describe la implementación de un protocolo de enrutamiento llamado árbol doble, el cual está basado en una topología de árbol y garantiza que cada nodo tenga dos rutas disjuntas en nodos hasta la raíz. El árbol doble emplea los mismos mensajes de configuración que se requieren para construir un árbol típico, disminuyéndose así el overhead en comparación con H-SPREAD [13] y DACA [14]. Además, garantiza que la red sea *k-connected* para k=2.

3 IMPLEMENTACIÓN DE PROTOCOLOS DE ENRUTAMIENTO

3.1 ENRUTAMIENTO BASADO EN TOPOLOGÍA DE ÁRBOL

La topología de árbol más sencilla de construir es aquella en la que cada nodo tiene un padre al cual enviar los datos medidos y retransmitir los datos recibidos. De esta forma, cada nodo envía sus datos a la raíz a través de múltiples saltos en la red. Generalmente, el padre de un nodo está más cerca a la raíz que el nodo mismo, de manera que cuando un nodo envía información a su padre, acerca el mensaje a su destino final, el cual es la raíz.

De acuerdo con [25] y [51], para construir un árbol simple de manera distribuida deben intercambiarse mensajes de configuración entre los nodos antes de enviar información de tráfico. Los mensajes de configuración tienen dos campos

fundamentales que son: La identidad del nodo que envía el mensaje (ID) y su distancia hasta la raíz. La distancia generalmente está dada en términos de una métrica, la cual depende de los requerimientos del sistema. Esa métrica puede ser alguna de las siguientes:

1. Número de saltos: Es el número de saltos que debe dar un paquete en la red para llegar a su destino. De acuerdo con esta métrica, la distancia entre cada par de nodos vecinos equivale a 1 salto. Minimizar el número de saltos equivale a minimizar el número de nodos por los que debe pasar un paquete para llegar a su destino.

2. Número de retransmisiones (EXT): Esta métrica expresa la confiabilidad de un enlace como el promedio de retransmisiones que se deben realizar para que un paquete se envíe exitosamente a través de ese enlace. Téngase en cuenta que la confiabilidad puede tomar un valor entre 0 y 1, de manera que, si la probabilidad de que una transmisión sea exitosa a través de un enlace es de p, entonces el EXT equivale a 1/p [43].

Es importante resaltar que previamente a la construcción de un árbol en una red, muchas veces se requiere una fase de pre-configuración en la que los nodos intercambian mensajes con sus vecinos para medir la calidad de los enlaces entre ellos. Esto es especialmente importante si la métrica de distancia se relaciona con la calidad en los enlaces, como es el caso del ETX. Asumiendo que esta fase ha concluido exitosamente, el envío de los mensajes de configuración para construir el árbol generalmente se ajusta a las siguientes reglas:

1. Inicialmente, la raíz tiene distancia 0 hasta la raíz. El resto de los nodos empieza con distancia infinita hasta la raíz. La raíz da inicio al proceso de construcción del árbol publicando su ID y su distancia en un mensaje de configuración.

2. Cuando un nodo recibe un mensaje de configuración, el nodo calcula su distancia a la raíz a través del remitente del mensaje de la siguiente manera: Si el remitente publica distancia K hasta la raíz, el nodo tendrá distancia D=K+A, donde A es la distancia entre los nodos, la cual ha sido hallada en la fase de pre-configuración. Luego, el nodo escoge al remitente como su padre y envía su propio mensaje de configuración publicando su ID y su distancia D.

Nótese que, de acuerdo con las reglas anteriores, un nodo no puede integrarse a la red ni enviar mensajes de configuración si no tiene al menos un padre.

Para garantizar la calidad en el servicio, muchas veces se requiere que los nodos envíen los datos de aplicación a través de la ruta más corta. Para ello, el árbol simple se construye mediante el algoritmo de Dijkstra corrido de manera distribuida. El algoritmo se ejecuta de acuerdo con las reglas anteriormente mencionadas, sin embargo, requiere que cada nodo actualice la identidad de su padre y su distancia a la raíz cuando reciba un mensaje de configuración de un nodo a través del cual haya una ruta más corta hasta la raíz. Además, el algoritmo requiere que cada vez que el nodo actualice su distancia, retransmita su mensaje de configuración. De esta forma, el árbol se va actualizando continuamente hasta que todos los nodos están conectados a la red a través de la ruta más corta.

En algunas ocasiones, los mensajes de configuración enviados por un nodo pudieran no llegar temporalmente a alguno de sus nodos vecinos debido a una falla de enlace temporal o a colisiones de los paquetes. Por tal motivo, algunas veces se requiere que los nodos envíen periódicamente su ID y su distancia a la raíz a sus nodos vecinos durante cierto tiempo o permanentemente. Cuando cada nodo actualiza y publica su estado con regularidad, el árbol simple puede calcularse continuamente, reparándose así ante fallas de nodos y enlaces. La principal desventaja de este cálculo continuo del árbol radica en que podría verse afectada la calidad del servicio debido a congestión en el tráfico cuando no hay fallas, dado que el envío de datos de aplicación debe convivir permanentemente

con el intercambio de mensajes de configuración. Además, también se ve afectado el tiempo de vida de la red, ya que los nodos deben intercambiar muchos mensajes de configuración adicionales a los datos de aplicación.

La ventaja del cálculo continuo del árbol radica en que, ante casos de fallas de nodos, siempre se encontrarán rutas para cada nodo vivo hasta la raíz, siempre y cuando la ruta exista. En vista de esto, se tiene aquí un intercambio entre resiliencia y calidad del servicio, ya que entre más frecuentemente se calcule el árbol, mejor será la resiliencia de la red en detrimento de la calidad del servicio. Para lograr un equilibrio entre ambos aspectos, algunos algoritmos como Collection Tree Protocol (CTP) han optado por construir un árbol donde la frecuencia de envío de los mensajes de configuración va cambiando con el tiempo.

De acuerdo con el protocolo CTP, el espacio de tiempo entre el envío de dos mensajes de configuración por parte de un mismo nodo viene dado por el algoritmo de Trickle. El algoritmo de Trickle básicamente indica que la frecuencia de envío de mensajes de configuración debe ser muy alta al inicio de la fase de configuración, e ir descendiendo paulatinamente a medida que pasa el tiempo y la topología tiende a estabilizarse.

3.2 PROTOCOLO BASADO EN ÁRBOL DE LA RUTA MÁS CORTA (SPT)

El primer protocolo basado en una topología de árbol que se implementó construye un árbol donde cada nodo tiene un solo padre a través del cual enviar los datos de aplicación. En esta implementación, cada nodo está conectado a la raíz a través de la ruta más corta. La implementación realizada es una adaptación del algoritmo descrito en [51]. Los mensajes de configuración tienen dos campos fundamentales que son: La identidad del remitente y su distancia hasta la raíz. La métrica empleada fue el número de saltos hasta la raíz. Se ha asumido que los enlaces entre dos nodos que son vecinos tienen confiabilidad del 100%. Así mismo, se ha asumido un modelo de transmisión ideal, es decir, que cada nodo tiene un círculo

a su alrededor de radio r en el cual todos los nodos que estén allí escuchan los mensajes con un 100% de probabilidad. Fuera de ese círculo se asume que la confiabilidad en las transmisiones es del 0%.

Para implementar el protocolo SPT, se implementó el algoritmo de Dijkstra de manera distribuida siguiendo las siguientes reglas:

1. Inicialmente, la raíz envía un mensaje de configuración publicando su identidad y distancia igual a 0.

2. Cuando un nodo recibe un mensaje de configuración, el nodo calcula su distancia a través de ese remitente. Sea d la distancia publicada por el remitente, la distancia a través de él será d+1.

3. Cuando un nodo envía su primer mensaje de configuración, envía otros nueve mensajes más esperando un tiempo T entre un mensaje y otro. Se ha establecido T como una variable aleatoria uniformemente distribuida entre 0 y 1 segundo para minimizar la posibilidad de que haya colisiones. En caso de que se presente una colisión, los nodos esperan a que les corresponda el turno de enviar el siguiente mensaje de configuración.

4. Siempre que un nodo recibe un mensaje de configuración de un nodo que publica menor distancia que su padre actual, el nodo cambia de padre y actualiza su distancia. Cuando al nodo le corresponda enviar su siguiente mensaje de configuración, publicará su distancia actualizada, donde dicha distancia se calcula de acuerdo con lo explicado en la regla 2.

En total, cada nodo transmite 10 mensajes de configuración, lo cual se ha establecido así para dar tiempo a que el cálculo del árbol converja. Así, una vez finalizado el cálculo, todos los nodos tienen la ruta más corta hasta la raíz. De esta forma, se tiene un árbol simple que se calcula continuamente en su fase inicial y en el que luego de un tiempo, se suspende el envío de mensajes de configuración.

La figura 2 muestra una red de 20 nodos. Dicha red se ha organizado en una topología de árbol simple de acuerdo con las reglas mencionadas en esta sección. El nodo 1 representa la raíz. Las líneas delgadas indican enlaces en los que la confiablidad en las transmisiones es del 100%. Por otra parte, las flechas gruesas indican los enlaces que se han establecido como parte de la topología luego de construirse el árbol.

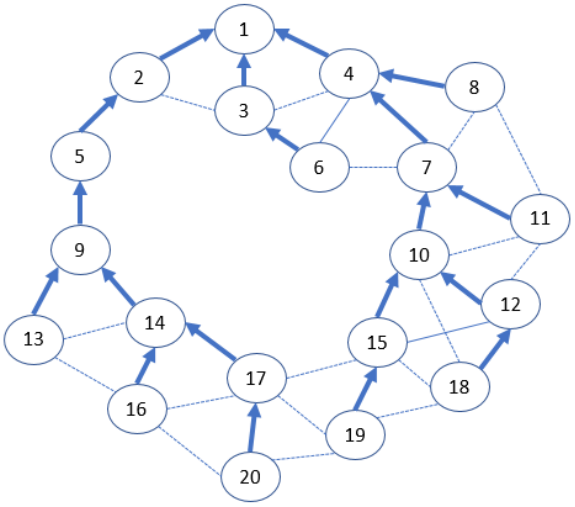

Figura 2. Red organizada en una topología de árbol simple.

Como se ha podido notar, la topología de árbol simple es muy sencilla de construir y no requiere que los nodos conozcan las rutas completas por las que pasa el paquete que envían. De hecho, los nodos solo necesitan conocer información local, a saber, la identidad del padre al cual enviar los datos que deben llegar a la raíz. Por tal motivo, el enrutamiento es bastante sencillo, ya que sigue la siguiente regla: *todo nodo debe dirigir hacia su padre el tráfico cuyo destino sea la raíz.*

La debilidad de la topología de árbol descrita es que no es muy resistente a fallos de nodos. En la figura 2 se evidencia claramente que, si un nodo falla, todos sus hijos quedan desconectados de la red, lo cual afecta negativamente el tiempo de vida, la cobertura y la confiabilidad en las transmisiones. Es por ello que se han ideado estrategias de enrutamiento para garantizar la resiliencia y la tolerancia a

fallos. La estrategia más importante para estos casos en los que se requiere elevar la tolerancia a fallos es el enrutamiento multicamino, el cual consiste en buscar y hacer uso de varias rutas disjuntas entre el nodo fuente y el nodo destino. Así, si un nodo falla, solo se ven afectadas las rutas que incluyen ese nodo, de manera que pueden emplearse rutas alternativas que se hayan encontrado en la fase de descubrimiento de rutas del protocolo.

La mayor parte de las estrategias de enrutamiento multicamino, se concentran en hallar múltiples rutas disjuntas entre un nodo fuente y un nodo destino específicos, no entre muchos nodos fuente y un nodo destino [13]. Sin embargo, el patrón de comunicación más usual en aplicaciones de recolección de información es el de muchos a uno. Por tal motivo, se hace necesario disponer de una estrategia de enrutamiento multicamino que tenga en cuenta este patrón de comunicación, el cual es la base de la topología de árbol. H-SPREAD tiene en cuenta estos aspectos [13], sin embargo, propone una estrategia en la que, para encontrar múltiples rutas disjuntas en nodos desde cada nodo hasta la raíz, deben intercambiarse una cantidad de mensajes de configuración que es superior a los que se intercambian en la construcción del árbol simple. Además, H-SPREAD tiene problemas de escalabilidad, dado que los mensajes de configuración contienen rutas completas, y no solamente el ID y la distancia, como es el caso en la construcción de un árbol simple [13]. El protocolo que se propone a continuación intenta proporcionar múltiples rutas disjuntas desde cada nodo de la red hasta la raíz utilizando el mismo número de mensajes de configuración que se usarían para la construcción de un árbol simple, sin necesidad de que los nodos conozcan las rutas completas que deben seguir los paquetes para llegar a la raíz.

3.3 DESCRIPCIÓN DE PROTOCOLO DE ENRUTAMIENTO DE ÁRBOL DOBLE

En el enrutamiento de árbol simple, cada nodo tiene un padre y una distancia hasta la raíz. Para mejorar la resiliencia de este esquema de enrutamiento básico, se impuso un nuevo requerimiento, el cual se resume así: *Para que un nodo pueda*

integrarse a la red, debe encontrar al menos dos padres, de lo contrario, el nodo no podrá enviar mensajes de configuración para seguir extendiendo el árbol. *Los únicos nodos que pueden omitir esta regla son los vecinos de la raíz, los cuales pueden extender el árbol teniendo de padre solamente a raíz, y la raíz misma, que no necesita tener padre para seguir extendiendo el árbol.* Este tipo de enrutamiento es llamado *árbol doble*. La regla básica consiste en que cada nodo debe encontrar al menos dos padres antes de seguir extendiendo el árbol desde la raíz hacia los extremos de la red. Es posible probar que, si cada nodo tiene dos padres, existen entonces al menos dos rutas disjuntas en nodos desde cada nodo hasta la raíz, de tal forma que se puede garantizar una red que sea k-connected con k=2. La demostración antes mencionada aparece en la sección de resultados.

La consecuencia de que existan dos rutas disjuntas en nodos desde cada nodo de la red hasta la raíz es que si un nodo en la red falla, todos los demás nodos quedan conectados a la red, sin importar qué nodo falló. El nodo que falla podría ser padre de muchos nodos, pero el hecho de que cada nodo tenga dos padres permite a sus nodos hijos desviar el tráfico a un padre alternativo para mantener la conectividad, sin incrementarse el retardo debido a una actualización de la ruta. Luego de la falla, la red podría quedar localmente más débil en términos de su resiliencia, debido a que algunos nodos quedarán ahora con un solo padre. Sin embargo, la resiliencia podría recuperarse mediante el intercambio de mensajes de configuración entre los nodos. La ventaja del árbol doble con relación a este punto es que el tiempo de recálculo de las rutas no sería crítico, ya que mientras el recálculo se realiza, continuaría habiendo al menos una ruta desde cada nodo de la red hasta la raíz. Lo anterior se asume verdadero siempre y cuando el proceso de recálculo se lleve a cabo antes de que se produzca la siguiente falla de algún nodo y la densidad de nodos permita configurar nuevamente un árbol doble. La figura 3 muestra una red configurada en una topología de árbol doble. El nodo 1 representa la raíz.

Para visualizar lo explicado anteriormente, obsérvese la red mostrada en la figura 2. Si en esta red falla el nodo 5, los nodos 9, 13, 14, 16, 17 y 20 se desconectan automáticamente de la red. Esta es la principal desventaja del protocolo SPT donde cada nodo tiene un solo padre a través del cual enviar datos hasta la raíz. Sin embargo, obsérvese ahora la red mostrada en la figura 3. En esta red configurada en la topología establecida por el árbol doble, cada nodo posee dos padres, incluyendo el nodo 5. Por tanto, si se produce una falla única de nodo, todos los demás nodos quedarán conectados a la red. Por tanto, se puede afirmar que el protocolo de árbol doble refuerza la resiliencia de la red ante fallos de nodos.

Al comparar las figuras 2 y 3 se puede observar también la principal desventaja del árbol doble con relación al árbol de un solo padre. En el árbol doble, las rutas de los nodos inicialmente son más largas, lo cual probablemente incrementa los retardos en las transmisiones de los paquetes. Por tanto, se puede observar que el árbol doble aumenta la resiliencia de la red, a la vez que incrementa los retardos en los paquetes durante la fase inicial de operación en la que ningún nodo ha fallado aún, de manera que se tiene un intercambio entre resiliencia de la red y retardos en los paquetes.

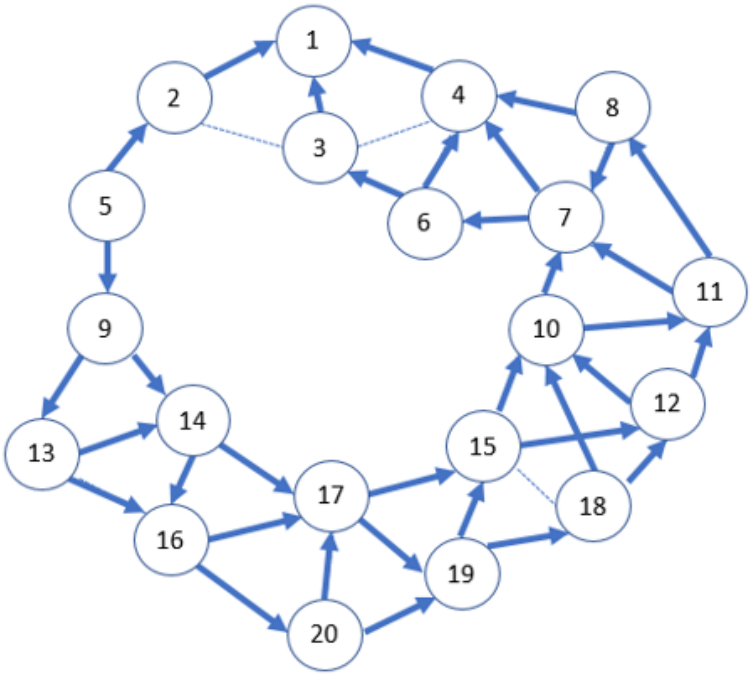

Figura 3. Red organizada en una topología de árbol doble.

3.3.1 Métricas usadas en el protocolo de árbol doble

En el protocolo de enrutamiento de árbol doble, cada nodo emplea información disponible de su vecindario para construir gradualmente el árbol desde la raíz hacia la periferia de la red. El proceso de construcción se lleva a cabo de manera similar a la versión distribuida del algoritmo de Dijkstra, de manera que cada nodo actualiza continuamente su distancia hasta la raíz y selecciona dos padres en cada actualización. Cada enlace entre dos nodos tiene un valor determinado de distancia. El objetivo es minimizar la distancia de cada nodo hasta la raíz, como es lo usual en el algoritmo de Dijkstra. Las métricas que se proponen para distancia son las siguientes:

1) Número de saltos. La distancia entre dos nodos vecinos equivale a 1. Minimizar la distancia en saltos equivale a minimizar el número de nodos que deben reenviar un mismo paquete antes de que llegue a la raíz.

2) Número esperado de transmisiones (ETX). Esta métrica expresa la confiabilidad de un enlace (la cual toma un valor entre 0% y 100%) como una función del número esperado de transmisiones que se necesitan para que un paquete logre llegar a la raíz. Si la probabilidad de que una transmisión a través de un enlace sea exitosa equivale a p, entonces la distancia en EXT para ese enlace equivale a 1/p [43].

3.3.2 Definición de distancia en el protocolo de árbol doble

En el protocolo de árbol doble, cada nodo debe tener al menos dos padres antes de seguir extendiendo el árbol. Un padre se define como un nodo que está más cercano a la raíz y al cual se envían los mensajes propios o provenientes de nodos hijos. Dado que cada nodo tiene dos padres, estos deben ser ordenados, de manera que se priorice a alguno de estos al momento de enviar la información. El primer padre tendrá entonces prioridad al momento de realizarse el envío de los mensajes. Cuando ese padre falle, los mensajes se desviarán al segundo padre, sin necesidad de que el árbol deba recalcularse.

Ahora se describirá la distancia empleada en el árbol doble. El hecho de tener dos padres significa que cada nodo recibe dos valores de distancia. Basándose en esos valores, el nodo debe entonces formular una distancia resultante que es la que finalmente propaga hacia sus vecinos para seguir extendiendo el árbol. Ese valor único de distancia se obtiene como un valor de peor caso (distancia más grande $\mathrm{D}_{\mathrm{i,j}}^{\mathrm{WC}}$), el cual garantiza que cada nodo esté más lejos que sus padres en el árbol. Las letras WC corresponden a una abreviación de “worst-case”, lo cual indica que $\mathrm{D}_{\mathrm{i,j}}^{\mathrm{WC}}$ corresponde a una distancia de peor caso. A continuación, se presenta cómo cada nodo calcula su distancia hasta la raíz luego de haber escogido a sus dos padres. Así mismo, se presentan los criterios bajo los cuales cada nodo escoge sus dos padres. Para ello, se empleará la siguiente notación:

$\mathrm{N}^{\mathrm{ALL}}$	conjunto de todos los nodos de la red.
$\mathrm{d}_{\mathrm{i,j}}$	distancia del nodo i al nodo j de acuerdo con la métrica empleada. Si $i \neq j \Leftrightarrow \mathrm{d}_{\mathrm{i,j}} > 0$.
$\mathrm{N}(\mathrm{i})$	conjunto de vecinos del nodo i.

$M(i)$	conjunto de padres del nodo i
$D_{i,j}^{WC\ via\ m}$	distancia del nodo i al nodo j usando al padre m como siguiente salto en la ruta. Es igual a $D_{m,j}^{WC} + d_{i,m}$.

Para el caso particular de la raíz, su distancia hasta ella misma equivale a cero, por tanto, la raíz no tiene padres.

$$D_{sink,sink}^{WC} \equiv 0; \quad M(sink) = \{\ \}; \ (1)$$

Los vecinos de la raíz tienen una distancia que equivale al valor dado por la métrica desde la raíz hasta ellos. Además, estos nodos no necesitan tener dos padres para extender el árbol, ya que son la única excepción a esta regla.

$$\forall_i \in N(sink): \ D_{i,sink}^{WC} \equiv d_{i,sink}, \qquad M(i) \equiv \{Sink\} \ (2)$$

Todos los demás nodos en la red comienzan con distancia infinita hasta la raíz, por tanto, el conjunto de dos padres empieza vacío para cada uno de estos nodos. Además, deben actualizar continuamente su distancia hasta la raíz a medida que el árbol se va extendiendo.

$$\forall_i \in \{N^{ALL} - Sink - N(Sink)\}: \ D_{i,sink}^{WC} = \infty, \qquad M(i) = \{\ \}; \quad (3)$$

Para que se compute el árbol doble, se requiere que cada nodo anuncie periódicamente su distancia hasta la raíz. Sea el nodo i, este actualiza su distancia basándose en la distancia $D_{i,sink}^{WC}$ anunciada por cada uno de sus vecinos j ∈ N (i), haciéndose así uso de una variante de la regla del algoritmo de Dijkstra de la ruta más corta.

En primer lugar, las distancia desde el nodo i hasta la raíz a través de cada uno de sus vecinos, es calculada así:

$$\forall_j \in N(i): D_{i,sink}^{WC\ via\ j} = D_{j,sink}^{WC} + d_{i,j} \quad (4)$$

A continuación, los vecinos de i son ordenados en una lista N^{LIST}, donde el siguiente nodo de la lista ofrece una distancia hasta la raíz que es igual o superior a la distancia que ofrece el nodo anterior, de acuerdo con lo calculado empleando la ecuación (4).

$$N^{LIST}(i) = \{j_1, j_2, \dots, j_{|N(i)|}\} \quad (5)$$

donde para cada j_n y $j_{n+1} \in N^{LIST}(i)$:

$$D_{i,sink}^{WC\ via\ j_n} \leq D_{i,sink}^{WC\ via\ j_{n+1}} \quad (6)$$

Esta lista se reduce descartando los nodos cuya distancia a la raíz sea infinita. Por tanto, la lista se restringe solamente a aquellos nodos que tienen distancia finita hasta la raíz y que por tal motivo están incorporados al árbol:

$$N^{CONN}(i) = N^{LIST}(i) - \{j: D_{i,sink}^{WC} = \infty\} = \{j_1, j_2, \dots, j_{|N^{CONN}(i)|}\} \quad (7)$$

En el algoritmo de Dijkstra que computa SPT, se toma únicamente el primer elemento j_1 de la lista como el nodo padre al cual el nodo i envía sus datos de aplicación, de manera que el nodo i queda con una distancia hasta la raíz que equivale a $D_{j_1,sink}^{WC} + D_{i,j_1}$. El resto de los nodos de la lista es descartado. La principal novedad del enrutamiento multicamino de árbol doble es que este selecciona los dos primeros elementos de la lista N^{LIST} como padres del nodo i, no solamente el primero:

$$M(i) \equiv N^{TOP}(i) = \{j_1, j_2\} \quad (8)$$

$M(i)$ es un conjunto, de manera que sus elementos no están ordenados. Es posible que un nodo tenga pocos vecinos, de manera que su lista $M(i)$ tenga solo un

elemento. En tal caso, a diferencia de lo que ocurre en el algoritmo de Dijkstra que construye un árbol típico, el nodo i tendrá una distancia $D_{i,sink}^{WC} = \infty$, de manera que no extiende el árbol.

Cuando $M(i)$ tiene ya dos elementos, la distancia del nodo hasta la raíz se actualiza de la siguiente manera:

$$D_{i,sink}^{WC} = \max_{j \in N^{TOP}(i)} \{D_{i,sink}^{WC \text{ via } j}\} = D_{i,sink}^{WC \text{ via } j_2} \quad (9)$$

Donde j_2 equivale al nodo vecino a través del cual hay una distancia $D_{i,j}^{WC}$ más larga hasta la raíz entre aquellos que hacen parte de la lista $N^{TOP}(i)$. Lo anterior quiere decir que la distancia del nodo i hasta la raíz a través del nodo j_2 (que es la que se ha escogido finalmente como distancia $D_{i,sink}^{WC}$) es más larga que la distancia desde i hasta la raíz a través de j_1. La razón por la que se ha decidido lo anterior es la siguiente. $D_{i,sink}^{WC}$ es la distancia más larga posible que separa al nodo i de la raíz, tomando en cuenta las distancias ofrecidas por los nodos de la lista $N^{TOP}(i)$. Aunque la escogencia de este valor como distancia que se propaga en el árbol parece ir en contra de la intuición, hay que tener en cuenta que la ruta con esta distancia no es necesariamente la ruta a través de la cual se enviarán los mensajes hasta la raíz. Para enviar los mensajes, se utilizará una ruta más corta, como se explicará más adelante. Por tanto, la ruta de longitud $D_{i,sink}^{WC}$ más bien representa la ruta más larga posible que podrá tener el nodo i antes de que se desconecte de la red debido a la presencia de fallos de nodos.

El enrutamiento de árbol doble intenta minimizar el valor de distancia $D_{i,sink}^{WC}$ para un nodo cualquiera i en la red, donde i siempre escoge los dos padres j_1 y j_2 que ofrece las menores distancias $D_{i,sink}^{WC \text{ via } j_1}$ y $D_{i,sink}^{WC \text{ via } j_2}$ respectivamente. Por otra parte, esta distancia cumple con propiedades importantes como las siguientes:

1) $\forall_i: D_{i,sink}^{WC} > 0$ (no negatividad)

2) $D_{i,sink}^{WC} = 0 \Leftrightarrow i = sink$ (identidad)

3) Las relaciones padre-hijo inducen un grafo dirigido en la red, con la raíz como destino final para cada uno de los nodos. Además, los nodos hijos siempre están más lejos de la raíz que sus respectivos padres.

4) La suma de la distancia del nodo i al nodo j con la distancia del nodo j hasta la raíz, es más larga que la distancia del nodo i hasta la raíz: $D_{i,j}^{WC} + D_{j,sink}^{WC} \geq D_{i,sink}^{WC}$ (desigualdad triangular).

Por tanto, D^{WC} es una métrica apropiada, donde cada nodo va actualizando continuamente su valor hasta que finalmente este converge a un valor mínimo. Sin embargo, debe establecerse un criterio que, una vez calculado el árbol doble, permita a los nodos seleccionar el padre (sea j_1 o j_2) al cual enviar los datos de aplicación.

3.3.3 Ordenando los padres

La selección del mejor padre debe seguir el principio de minimización de la longitud de la ruta que un paquete debe recorrer para llegar a la raíz. Por tanto, se debe incluir otro cómputo de distancia en las rutas multicamino del árbol doble desde un nodo i hasta la raíz. Esta nueva distancia obedece las reglas del protocolo SPT, pero está restringida al gráfico dirigido que resulta luego de la computación del árbol doble. Esta distancia será llamada distancia implícita más corta (D^{SE}), y permite ordenar el conjunto de dos padres en cada nodo a fin de que cada nodo sepa a qué padre enviar los datos de aplicación. Las letras SE corresponden a una abreviación de “Shortest embedded”, lo cual indica que D^{SE} corresponde a la distancia más corta del nodo hasta la raíz teniendo en cuenta solamente los enlaces establecidos por la topología de árbol doble. Finalmente, la distancia final

desde cada nodo hasta la raíz corresponderá entonces a la distancia más corta posible que puede resultar empleando los enlaces de la topología establecida por el árbol doble.

Las reglas para calcular D^{SE} son las mismas que aquellas empleadas por el algoritmo de Dijkstra. Cada nodo i debe anunciar periódicamente su distancia D_i^{SE} hasta la raíz y actualizarla con base en la información publicada por sus vecinos. Inicialmente, la raíz anuncia su distancia:

$$D_{sink}^{SE} \equiv 0;\ M(Sink) = \{\quad\} \quad (10)$$

De manera similar, los vecinos de la raíz $\forall_i \in N(Sink)$ anuncian su distancia y escogen como padre a la raíz misma.

$$D_i^{SE} \equiv d_{i,sink}; M(i) \equiv Sink \quad (11)$$

Los demás nodos tienen las siguientes condiciones iniciales:

$$D_i^{SE} \equiv \infty;\ M_i^{LIST} \equiv \{\quad\} \quad (12)$$

Ahora bien, la distancia D_i^{SE} desde el nodo i hasta la raíz a través de sus padres es obtenida así:

$$\forall_m \in M(i): D_i^{SE,\ \text{via m}} = D_{m,Sink}^{SE} + d_{i,m} \quad (13)$$

A continuación, los padres $|M(i)|$ de i son ordenados en una lista:

$$M^{LIST}(i) = \{m_1, m_2\} \quad (14)$$

donde $|M(i)| = 2$, y además:

$$D_i^{SE \text{ via } m_1} \leq D_i^{SE \text{ via } m_2} \quad (15)$$

Finalmente, la distancia D^{SE} desde el nodo i hasta la raíz, es el valor más pequeño disponible:

$$D_i^{SE} = \min_{m \in M(i)} \{D_i^{SE,\text{via } m}\} = D_i^{SE,\text{via } m_1} \quad (16)$$

En los pasos anteriores se evidencia que m_1 es el padre que ofrece la ruta más corta hasta la raíz de entre los dos padres seleccionados. Por tanto, m_1 es el nodo al cual i envía los datos de aplicación. Cuando m_1 falla, los mensajes se desvían a m_2, el cual es escogido como padre alternativo.

3.3.4 Escalabilidad del protocolo de árbol doble

La escalabilidad comprende tanto el overhead del protocolo como la cantidad de variables de estado que este utiliza. A continuación, se analiza el protocolo de árbol doble en término de estos dos aspectos:

1) *Overhead del Protocolo*. Cada nodo debe anunciar periódicamente el valor de su distancia hasta la raíz, tanto su distancia larga $D_{i,sink}^{WC}$ como su distancia corta D^{SE}, sin importar el tamaño de la red. Sin embargo, en DACA [14] algunos mensajes de configuración incluyen las rutas completas del nodo hasta la raíz, de manera que el tamaño del mensaje aumenta con el tamaño de la red para el caso de algunos nodos.

2) Variables de estado. Cada nodo debe guardar las dos distancias mencionadas en el punto anterior, además de la identidad de sus dos padres. Estas cuatro variables deben almacenarse y actualizarse continuamente durante la fase de construcción del árbol. Además, cada nodo también debe conocer su distancia a

cada uno de sus vecinos de acuerdo con la métrica escogida, las cuales fueron presentadas en la sección 3.3.1.

3.3.5 Algoritmo del protocolo de árbol doble

El algoritmo del protocolo de árbol doble asume una red donde solo hay una raíz y cada nodo busca incorporarse por sí mismo a la topología de árbol. Cada nodo tiene una vista local de la red, representadas principalmente por las distancias entre los nodos de acuerdo con las métricas presentadas en la sección 3.3.1 y la información recolectada progresivamente de sus vecinos. El algoritmo requiere que cada nodo publique periódicamente dos distancias: Una para computar el árbol doble y establecer las rutas desde cada nodo hasta la raíz, y otra que permite ordenar los padres y establecer aquel que ofrece la ruta más corta hasta la raíz, donde dicha ruta solamente puede emplear los enlaces establecidos por la topología de árbol doble que se ha construido.
Las ecuaciones (1–3) y (10–12) marcan el estado inicial de cada nodo, tanto la distancia empleada para escoger los padres como la distancia de la ruta más corta. La raíz es un nodo especial, ya que se asume que nunca falla. Los vecinos de la raíz también son nodos especiales, ya que no deben tener dos padres para seguir extendiendo el árbol. El resto de los nodos ejecuta lo descrito en las ecuaciones (4–9) y (13–16) una vez que reciben información proveniente de sus vecinos. Con relación a la forma en que se ejecuta este último punto, existen dos alternativas, que son:

1) La información sobre el vecindario es recolectada y actualizada en una tabla dentro del nodo. Una vez que esto ocurre, el nodo actualiza la información y publica su mensaje de configuración haciendo uso de la información de la tabla.

2) Una vez que se recibe información de un nodo vecino, todos los nodos que escuchan el mensaje actualizan su información. Sin embargo, cada nodo publica su mensaje de configuración cada intervalo regular de tiempo T basándose en la información actualizada.

3.4 COMPARACIÓN ENTRE EL ÁRBOL SIMPLE Y EL ÁRBOL DOBLE

Como se ha descrito hasta aquí, el algoritmo que construye un árbol simple y el algoritmo que construye un árbol doble en una red inalámbrica de sensores son diferentes. En la construcción del árbol doble, los requerimientos que debe cumplir un nodo para seguir extendiendo el árbol son más exigentes, ya que, para extender el árbol, un nodo debe tener al menos dos padres que le proporcionen dos rutas disjuntas hasta la raíz, a excepción de los nodos que son vecinos de la raíz. Estos últimos pueden extender el árbol teniendo de padre únicamente a la raíz. Esta diferencia hace que tanto el árbol simple como el árbol doble luzcan de una forma distinta, ya que crecen de manera diferente. Es de esperarse que el árbol doble tienda a crecer hacia donde hay mayor densidad de nodos, de manera que cada nodo quede conectado a la raíz a través de una zona lo suficientemente densa como para que la red satisfaga el requerimiento de resiliencia en el que cada nodo tiene dos rutas disjuntas hasta la raíz. Para observar más claramente este hecho, se tiene la red de la figura 4. En dicha red, las líneas azules indican los enlaces en los que puede establecerse comunicación. El nodo 1 representa la raíz. Recuérdese que se ha asumido que el modelo de trasmisión de los nodos es ideal, es decir, los enlaces tienen confiabilidad del 100% cuando la distancia entre dos nodos es menor que R, donde R es un número fijo. Por otra parte, cuando dos nodos están alejados una distancia mayor a R, se asume confiabilidad de 0% en el enlace. Para simplificar el modelo de transmisión, estas probabilidades no tienen en cuenta cómo afecta la congestión en el tráfico.

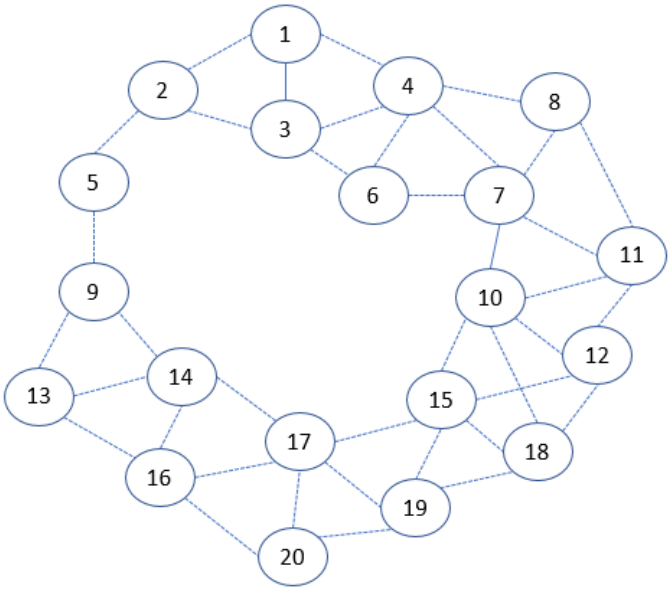

Figura 4. Red con todos los enlaces disponibles.

Inicialmente el nodo 1 transmite su propio mensaje de configuración. Cuando los nodos 2, 3 y 4 reciben dicho mensaje, cada uno de ellos establece a la raíz como su padre. Esto ocurre tanto en el árbol simple como en el árbol doble, tal como se observa en la figura 5.

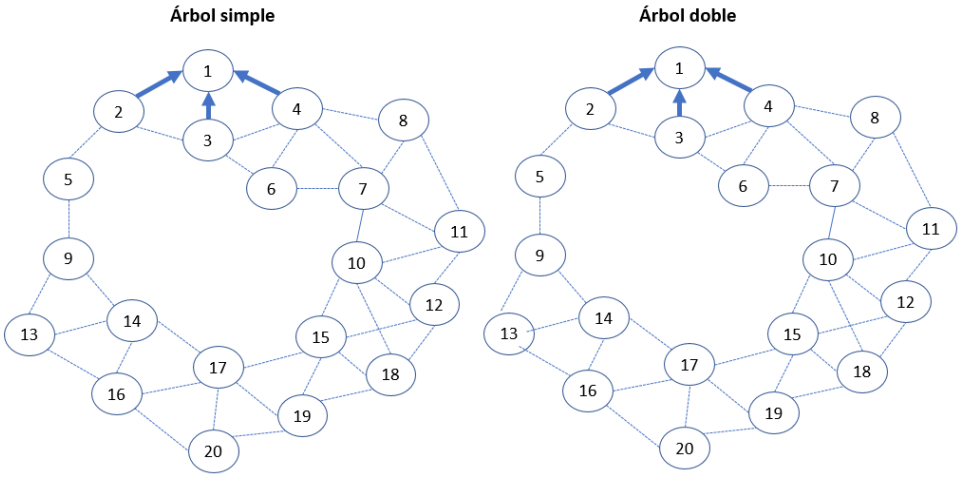

Figura 5. Árbol simple vs Árbol doble, paso 1.

En la siguiente fase, los nodos 2, 3 y 4 envían sus propios mensajes de configuración. En el árbol simple, el resultado es que el nodo 5 adopta como padre

al nodo 2, el nodo 6 podría adoptar al nodo 3 o al nodo 4, mientras que los nodos 7 y 8 adoptan como padre al nodo 4. En el árbol doble, el nodo 6 termina adoptando dos padres, a saber, el nodo 3 y el nodo 4, mientras que los nodos 5, 7 y 8 de momento solo tienen un padre. La figura 6 ilustra esta situación.

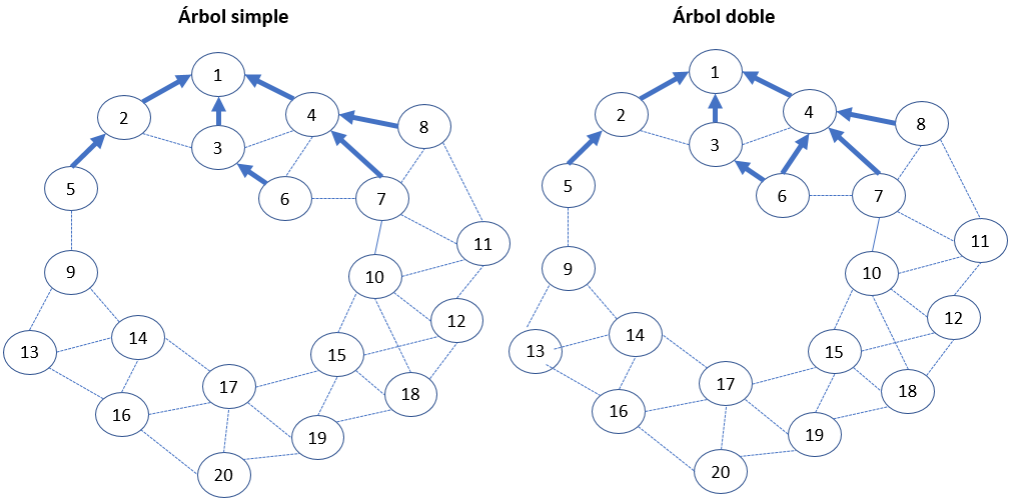

Figura 6. Árbol simple vs Árbol doble, paso 2.

De acuerdo con el protocolo de árbol simple, los nodos 5, 6, 7 y 8 están conectados a la red, de manera que envían sus propios mensajes de configuración (Véase la figura 6). Sin embargo, de acuerdo con el protocolo de árbol doble, de estos nodos solo el nodo 6 puede enviar mensajes de configuración, ya que es el único que tiene dos padres. En este hecho se evidencia la diferencia fundamental entre la forma en que crece el árbol simple y la forma en que crece el árbol doble.

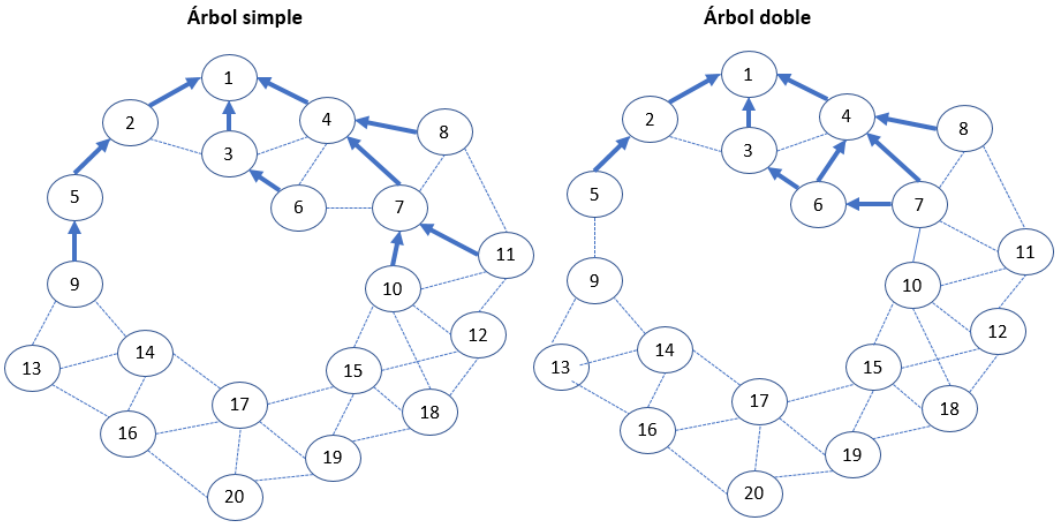

Figura 7. Árbol simple vs Árbol doble, paso 3.

En la figura 7 se observa claramente que, en el árbol doble, el nodo 7 adopta al nodo 6 como su padre cuando el nodo 6 envía su mensaje de configuración. Una vez que esto ocurre, el nodo 7 puede seguir extendiendo el árbol, ya que tiene dos padres, a saber, los nodos 4 y 6. En la figura 7 también se observa que, en el árbol doble, el nodo 5 no hace crecer el árbol debido a que solo tiene un padre. Sin embargo, en el árbol simple el nodo 5 sí hace crecer el árbol. Esto genera una debilidad en el árbol simple, ya que si el nodo 2 falla, tanto el nodo 5 como todos sus hijos quedan desconectados de la red. El hecho de que el nodo 5 no haga crecer el árbol en la topología de árbol doble, minimiza el número de nodos que quedarían desconectados en caso de que el nodo 2 falle.

Dado que el nodo 7 ahora tiene dos padres, envía su propio mensaje de configuración en la topología de árbol doble. Ese mensaje es recibido por los nodos 8, 10 y 11, los cuales adoptan al nodo 7 como uno de sus padres. Por otra parte, en la topología de árbol simple, los nodos 9, 10 y 11 envían sus propios mensajes de configuración para seguir extendiendo el árbol, dado que satisfacen el requisito de tener al menos un padre (Véase la figura 8).

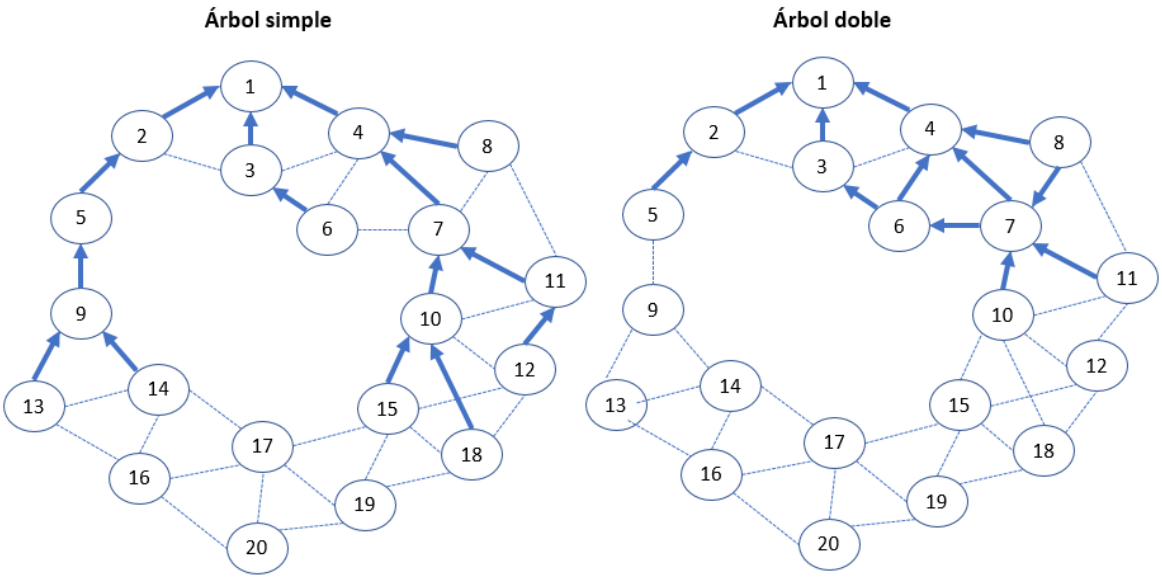

Figura 8. Árbol simple vs Árbol doble, paso 4.

Como se observa en la figura 8, en la topología de árbol doble el nodo 8 tiene dos padres, de manera que envía su mensaje de configuración para seguir extendiendo el árbol. Los nodos 10 y 11 no extienden el árbol, dado que solo tienen un padre. Por otra parte, en la topología de árbol simple, los nodos 12, 13, 14, 15 y 18 envían sus propios mensajes de configuración para seguir extendiendo el árbol, dado que tienen al menos un padre respectivamente. En la figura 9 se observa que luego de este proceso, en el árbol doble el nodo 11 adopta al nodo 8 como padre, satisfaciendo así el requisito de tener dos padres para seguir extendiendo el árbol. Así, el nodo 11 puede enviar su propio mensaje de configuración. Por otra parte, en el árbol simple, los nodos 16, 17 y 19 se conectan a la red.

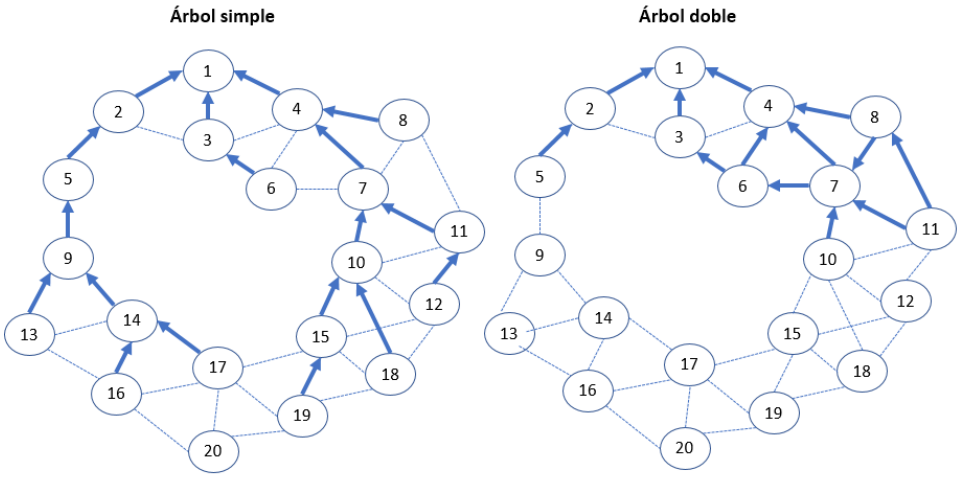

Figura 9. Árbol simple vs Árbol doble, paso 5.

Como se observa en la figura 9, los nodos 16, 17 y 19 pueden enviar sus propios mensajes de configuración en la topología de árbol simple, dado que tienen al menos un padre. El resultado de esto es que el nodo 20 se conecta a la red y adopta a alguno de estos nodos como su padre. Por otra parte, en el árbol doble el nodo 11 envía su mensaje de configuración para seguir extendiendo el árbol, dado que tiene dos padres. Ese mensaje de configuración enviado por el nodo 11, es recibido por los nodos 10 y 12, los cuales adoptan al nodo 11 como padre, tal como se observa en la figura 10.

En la figura 10, se observa que ha finalizado la construcción del árbol simple. Sin embargo, la construcción del árbol doble continúa. En la topología del árbol doble, se observa que el nodo 10 ya satisface el requisito de tener dos padres, de manera que ahora envía su mensaje de configuración para seguir extendiendo el árbol. Ese mensaje llega a los nodos 12, 15 y 18, los cuales adoptan al nodo 10 como su padre, tal como se observa en la figura 11.

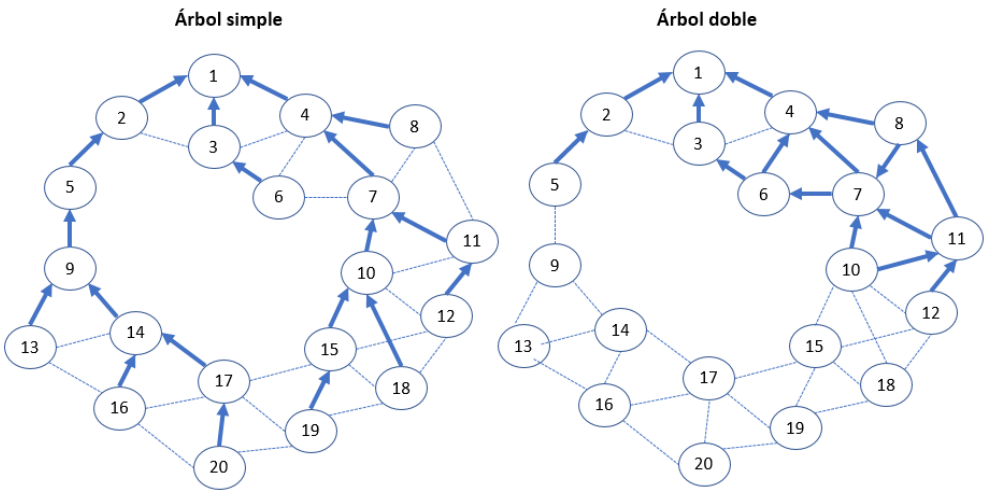

Figura 10. Árbol simple vs Árbol doble, paso 6.

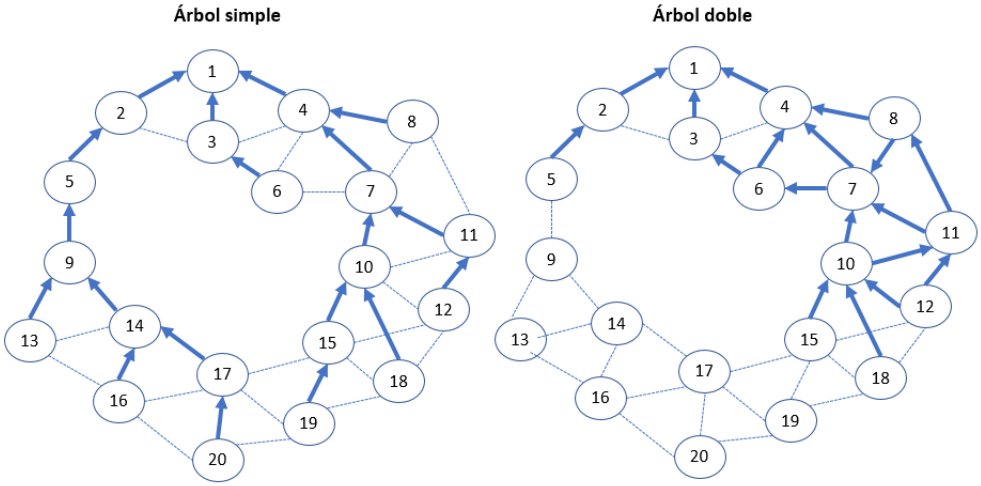

Figura 11. Árbol simple vs Árbol doble, paso 7.

En la figura 11 se observa que, en la construcción del árbol doble, el nodo 12 tiene dos padres, de manera que ya satisface los requisitos para continuar extendiendo el árbol. El mensaje de configuración del nodo 12 es recibido por los nodos 15 y 18, los cuales adoptan al nodo 12 como padre. A su vez, los nodos 15 y 18 quedan con dos padres, de manera que continúan extendiendo el árbol. Finalmente, la figura 12 muestra cómo queda la red una vez se han construido tanto el árbol simple como el árbol doble.

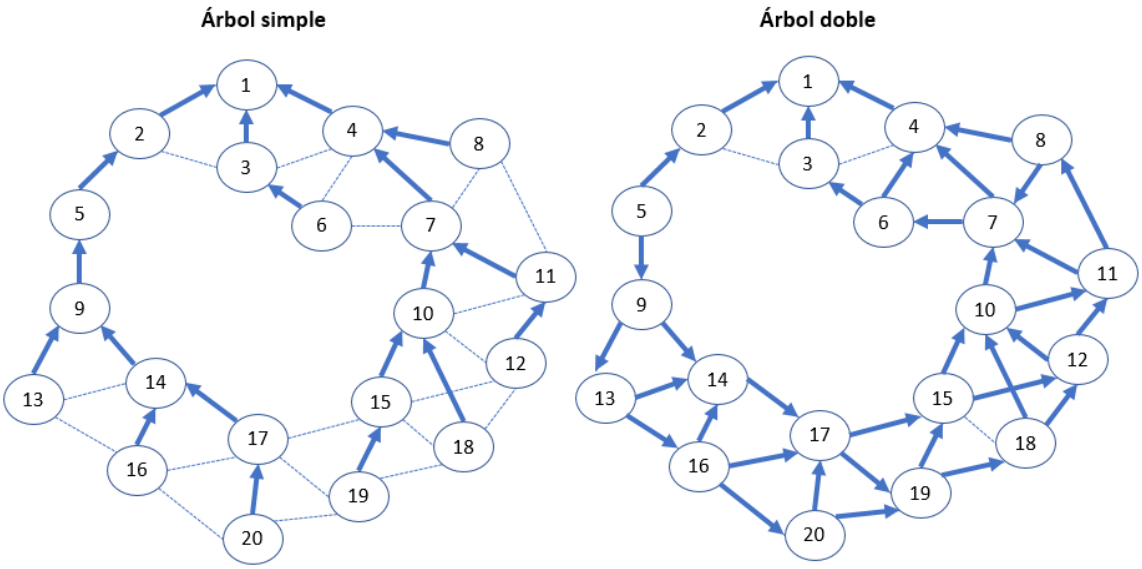

Figura 12. Árbol simple vs Árbol doble, paso 8.

Se pudo observar claramente que, en el árbol simple, el árbol crece más rápido que en el árbol doble. Sin embargo, en el árbol doble, cada nodo debe tener dos padres para seguir extendiendo el árbol, a excepción de los nodos que tienen a la raíz como padre. Este hecho hace que el árbol doble sea más resistente a fallas de nodos. Si en el árbol doble falla un nodo, se garantiza que todos los nodos restantes quedan conectados a la red, sin importar qué nodo falle, lo cual no puede ser garantizado por el árbol simple. Tómese como ejemplo lo observado en las figuras 13 y 14. En la figura 13 se observa que si el nodo 2 falla, se desconectan 7 nodos de la red además del nodo dañado. Sin embargo, en el árbol doble, una falla en el nodo 2 no desconecta ningún otro nodo de la red. Esto es así dado que, en la construcción del árbol doble, el nodo 5 no extiende el árbol hasta no tener dos padres, de manera que el árbol que va creciendo es resistente a fallos únicos de nodos. Igualmente, en la figura 14 se observa que cuando el nodo 5 falla, 5 nodos quedan desconectados en el árbol simple y ningún nodo aparte del que falla queda desconectado en el árbol doble.

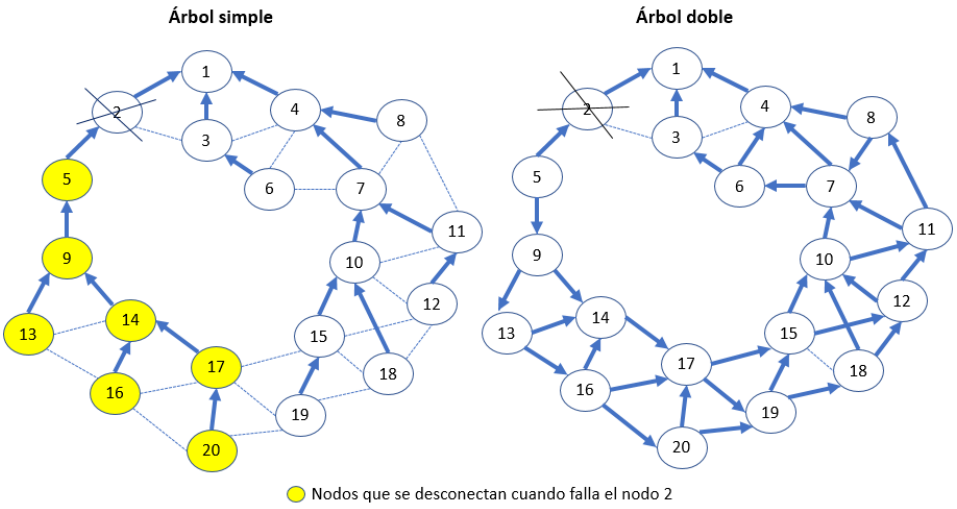

Figura 13. Árbol simple vs Árbol doble cuando falla el nodo 2.

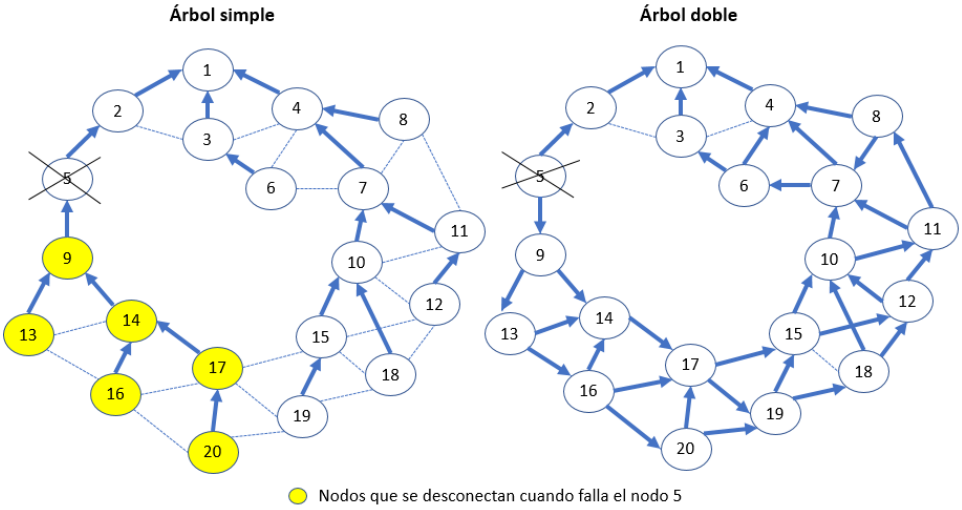

Figura 14. Árbol simple vs Árbol doble cuando falla el nodo 5.

4 RESULTADOS

4.1 DEMOSTRACIÓN DE TEOREMA

Como se ha mencionado en secciones anteriores, una de las principales fortalezas del árbol doble es que garantiza que cada nodo de la red tenga dos rutas disjuntas en nodos hasta la raíz. Este teorema procede a probarse a continuación.

Teorema. Una ruta multicamino de un nodo a otro donde cada nodo tiene dos padres tiene al menos dos rutas disjuntas en nodos hasta la raíz.

Demostración: La demostración se hará por contradicción. La ruta multicamino puede verse como un gráfico dirigido desde el nodo fuente hasta la raíz, donde las flechas siempre van en la dirección de hijo a padre. Ahora asúmase que cada nodo tiene dos padres, pero no hay dos rutas disjuntas en nodos hasta la raíz. Lo anterior implica que en algún nodo la ruta doble debe colapsar. La figura 15 ilustra esta situación, donde los nodos A y B son hijos del nodo C y la ruta multicamino colapsa en ese nodo, invalidándose así la existencia de dos rutas disjuntas.

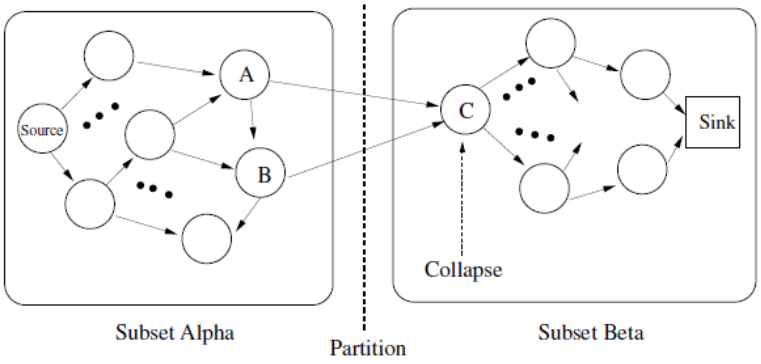

Figura 15. Partición del gráfico dirigido inducido por la ruta multicamino con un nodo donde la ruta multicamino colapsa.

En este punto es claro que los nodos que están a la derecha de C están más cerca a la raíz que el nodo C, y que los nodos a la izquierda de C están más lejos.

Considérese una partición del grafico dirigido que pasa por las conexiones que unen los nodos A y B con el nodo C. Los nodos en el gráfico multicamino dirigido quedan entonces divididos en dos conjuntos disjuntos: Alpha, en el cual están los nodos A, B y el nodo fuente, y Beta, el cual contiene el nodo C, sus padres, y la raíz.

De acuerdo con las suposiciones realizadas, tanto el nodo A como el nodo B deben tener dos padres, donde uno de ellos es el nodo C. El otro padre debe estar dentro del conjunto Alpha. Los padres de A y B están más cerca a la raíz que los nodos A y B respectivamente, lo cual se deriva de la definición de distancia $\mathrm{D}_{\mathrm{i,sink}}^{\mathrm{WC}}$, como se explicó en la sección 3.3.2.

Sea P el nodo más cercano a la raíz de entre los padres de A y B. El padre de P estará necesariamente en Alpha, ya que el único enlace que atraviesa la partición entre Alpha y Beta es aquella que va desde los nodos A y B hasta el nodo C. Los padres de P tendrán también padres en el conjunto Alpha, y así sucesivamente. Sin embargo, Alpha es finito, de manera que debe haber al menos un nodo en Alpha que no tenga padres en Alpha, contradiciéndose así la afirmación inicial de que la ruta multicamino colapsa en el nodo C. Por tanto, se concluye que existe al menos otra ruta simple dentro de la ruta multicamino que sea disjunta en nodos, con lo cual finaliza la demostración del teorema.

4.2 SIMULACIONES

4.2.1 Evaluación de resiliencia

El protocolo de árbol doble fue implementado en el simulador para redes Omnet++ 4.6 haciendo uso de la framework de MiXiM (v2.3) para modelar redes inalámbricas de sensores IEEE 802.15.4 de baja potencia. Actualmente, la framework de MiXiM se ha integrado con otra llamada INET, la cual está más orientada a interconectividad IP y cubre una mayor variedad de aplicaciones.

El algoritmo del árbol doble ha sido evaluado para redes aleatorias confinadas a un área cuadrada donde los nodos fallan en un orden aleatorio, y por cada falla de nodo, se verifica cuántos quedan conectados a la raíz. Se ha implementado tanto SPT como el protocolo de árbol doble para la misma cantidad de redes aleatorias. El árbol doble no se recalcula luego de que se ha configurado. Cuando el padre de un nodo falla, el nodo reacciona desviando los datos de aplicación a su padre alternativo de acuerdo con lo estipulado en $\mathrm{M}^{\mathrm{LIST}}$.

1) Parámetros:

Área: 150m x 150m
Número de nodos: 100, posicionados aleatoriamente
Radio de transmisión: 30m
Modelo de transmisión: Circular e Ideal
Estándar: IEEE 802.15.4
Métrica de distancia: Número de saltos
Módulo NIC de MiXiM: Nic802154_TI_CC2420

2) Escenarios:

Número de nodos dañados: 1, 2,: : : ,99
Algoritmo: SPT, árbol doble

3) Métrica: Número de nodos que permanecen conectados

4) Número de repeticiones: 50.

La primera simulación se realizó con el objetivo de comprobar el teorema demostrado en la sección 4.2. Para ello, se implementó tanto el protocolo de árbol doble como el protocolo de árbol SPT en 50 redes distintas, donde cada red estaba conformada por 100 nodos distribuidos aleatoriamente en un área cuadrada de acuerdo con las condiciones descritas anteriormente. Para cada red, se procedió a hacer fallar aleatoriamente 1 nodo (distinto a la raíz). Luego de esto, se procedió a verificar cuántos nodos permanecían conectados a la red de acuerdo con la lógica de funcionamiento de cada uno de los dos algoritmos implementados. Dado que cada red tiene 100 nodos, lo ideal es que cuando un nodo falla queden 99 nodos conectados a la red, de manera que el fallo del nodo no afecte la comunicación de ningún otro nodo con la raíz. Los resultados obtenidos fueron los siguientes:

- De acuerdo con el protocolo SPT, en el 38% de los casos un fallo aleatorio de un de nodo afectó la comunicación de otros nodos con la raíz.

- De acuerdo con el protocolo de árbol doble, en el 0% de los casos un fallo aleatorio de un nodo afectó la comunicación de otros nodos con la raíz.

Los resultados anteriores confirman que, en el árbol doble, efectivamente cada nodo tiene dos rutas disjuntas en nodos hasta la raíz, ya que un fallo aleatorio de un nodo en la red no afectó la comunicación de ningún otro nodo con la raíz. Mediante estos resultados, se corrobora que el protocolo de árbol doble garantiza que la red sea k-connected para k=2 en una red con un patrón de comunicación de muchos a uno.

A continuación, la evaluación tanto del protocolo de árbol doble como del protocolo SPT se realizó en simulaciones en Omnet++ como se describe a continuación: Primero se construía un árbol en la red siguiendo las reglas de cada protocolo respectivamente. Cuando un nodo fallaba, se verificaban cuantos nodos permanecían conectados a la red. Este procedimiento se repetía hasta que todos los nodos fallaban. Nunca fallaban dos nodos simultáneamente. Antes bien, antes de que el siguiente nodo fallase, se verificaba el número de nodos conectados a la red. De acuerdo con el protocolo ideal en términos de esta métrica, el número de nodos dañados siempre equivale al número de nodos desconectados de la red, de manera que una falla de nodo no afecta la comunicación de ningún otro nodo en la red. Sin embargo, generalmente cuando un nodo falla en una red configurada en topología de árbol, la conectividad de sus hijos se ve afectada, especialmente si los nodos hijos no cuentan con padres alternativos hacia los cuales desviar los datos de aplicación.

Las figuras 16 y 17 presentan la media de los resultados obtenidos incluyendo los intervalos de confianza del 90%.

Los resultados de la figura 16 muestran que, para los dos algoritmos evaluados, los intervalos de confianza no se traslapan, lo cual permite concluir los siguiente: En el árbol doble, la conectividad siempre es mayor que en el árbol típico de un solo padre para un número específico de nodos dañados. Por tanto, cuando el árbol crece siguiendo las reglas del protocolo de árbol doble, la resiliencia de la red se eleva considerablemente, sin que se incrementen los mensajes de configuración que se necesitan para construir el árbol.

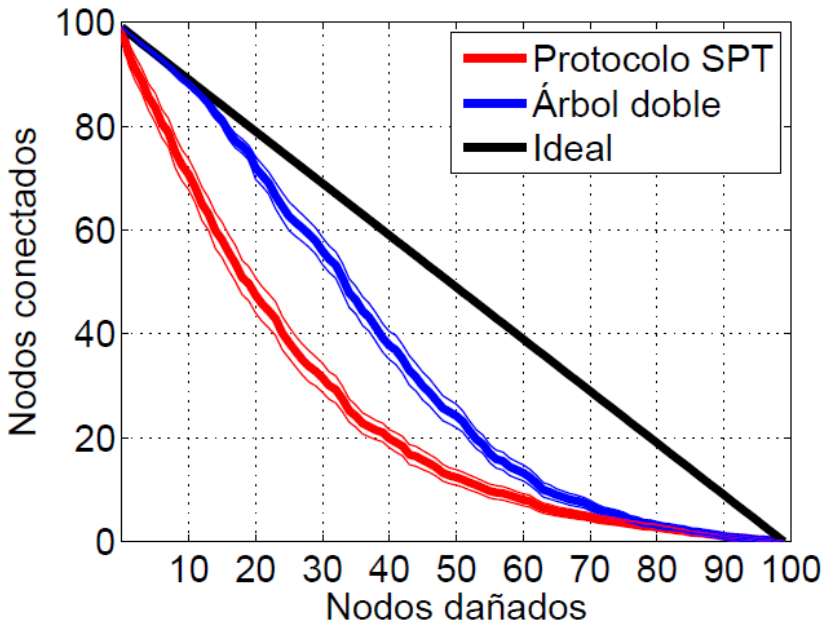

Figura 16. Comparación del protocolo SPT con el protocolo de árbol doble.

Los resultados también fueron graficados en término de nodos dañados en la red vs la proporción entre nodos conectados y nodos sobrevivientes en la red. Un nodo sobreviviente es un nodo que no ha fallado. Por otra parte, un nodo conectado es un nodo que aún tiene una ruta para mandar datos de aplicación a la raíz. El protocolo ideal en términos de la métrica mencionada es aquel en el que cuando un nodo falla, todos los nodos sobrevivientes quedan conectados a la red, de manera que la proporción entre nodos conectados y nodos sobrevivientes permanece en 1. Sin embargo, lo anterior no siempre ocurre, ya que cuando un nodo falla, posiblemente otros nodos (por ejemplo, sus hijos en el caso de una topología de árbol simple) que no presenten falla alguna se vean afectados y se desconecten de la red, especialmente si el nodo que ha fallado es empleado por muchos otros nodos como intermediario para enviar mensajes hasta la raíz. La figura 17 presenta la media de los resultados obtenidos con los intervalos de confianza del 90%.

Los resultados de la figura 17 muestran que los intervalos de confianza no se traslapan en el intervalo en el que fallan hasta el 70% de los nodos, lo cual permite

concluir lo siguiente: Con un 90% de confianza, en el árbol doble la proporción de nodos conectados vs nodos sobrevivientes en la red es superior a la misma proporción en el protocolo SPT, para un número específico de nodos dañados en la red. Lo anterior permite comprobar que el árbol doble incrementa significativamente la resiliencia de la red en comparación con el protocolo SPT que construye un árbol típico. Es deseable que la proporción entre nodos conectados y nodos sobrevivientes sea lo más grande posible, ya que esto significa que son pocos los nodos que son funcionales y no están siendo empleados por estar desconectados de la red a causa de fallos en otros nodos.

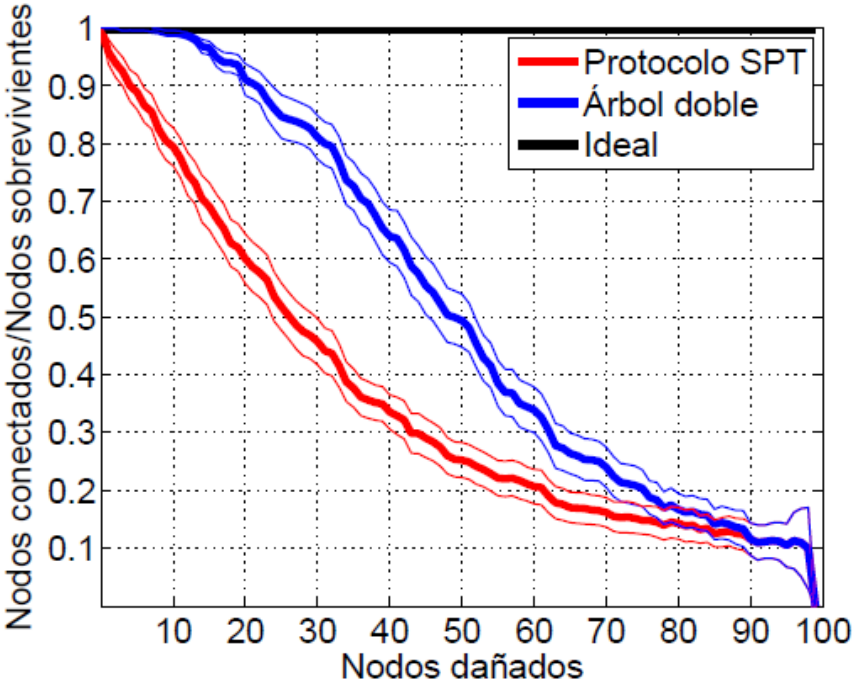

Figura 17. Comparación del protocolo SPT con el protocolo de árbol doble.

4.2.2 Evaluación de retardo

Como se pudo observar en la sección anterior, el protocolo de árbol doble resulta ser más resiliente que el protocolo SPT basado en un árbol sencillo cuando fallan progresivamente nodos en la WSN. Ahora obsérvese la figura 18, donde el nodo 1 es la raíz. Allí se muestra una red de 20 nodos configurada tanto en la topología estipulada por el árbol doble como en una topología basada en el protocolo SPT.

Tómese ahora de ejemplo al nodo 9. En el protocolo SPT basado en una topología de árbol simple, el nodo 9 está a 3 saltos de la raíz a través de los nodos 5 y 2 respectivamente. Esta es la ruta más corta posible entre los nodos 9 y 1. Ahora bien, en una topología de árbol doble, una de las rutas más cortas entre los nodos 9 y 1 es a través de los nodos 14, 17, 15, 10, 7 y 4, de manera que el nodo 9 está a 7 saltos de la raíz. Es imposible que exista una ruta de menor tamaño en el árbol doble para la red de la figura 20. Por tanto, tomando como ejemplo la red de la Figura 18, es posible observar que, si bien el árbol doble es más resiliente que el árbol simple, las rutas en el árbol doble tienden a ser más largas que en el árbol simple, lo cual podría reflejarse en un aumento del retardo en los paquetes enviados por los nodos. El objetivo de esta sección es precisamente verificar mediante simulaciones cuál es el incremento de los retardos en los paquetes de los nodos en el árbol doble, en comparación con el retardo de los paquetes enviados por los nodos en la topología de árbol simple.

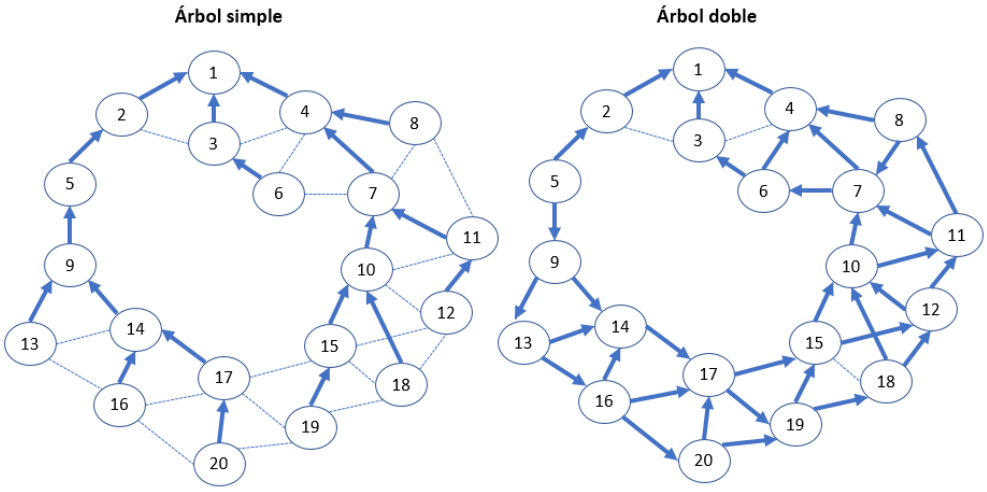

Figura 18. Árbol simple vs Árbol doble

A continuación, se describen los aspectos más importantes de las simulaciones realizadas:

1) Parámetros:
Área: 150m x 150m
Número de nodos: 100, posicionados aleatoriamente
Radio de transmisión: 30m
Modelo de transmisión: Circular e Ideal
Estándar: IEEE 802.15.4
Métrica de distancia: Número de saltos
Módulo NIC de MiXiM: Nic802154_TI_CC2420

2) Escenarios:
Algoritmo: SPT, árbol doble
Número de paquetes enviados por nodo (A excepción de la raíz): 30

3) Métrica: Retardo

4) Número de repeticiones: 30.

Para evaluar el incremento del retardo en *árbol doble* en comparación con SPT, cada nodo enviaba 30 mensajes hasta la raíz. Cada mensaje era enviado cuando la red estaba completamente libre de tráfico adicional a dicho mensaje. Luego, se calculó el promedio del retardo de los 30 mensajes enviados por cada nodo, obteniéndose así 99 valores de retardo promedio, un valor por cada nodo (Recuérdese que, de los 100 nodos de la red, uno de ellos corresponde a la raíz, la cual no envía datos de aplicación). Sea entonces N_i^S el valor de retardo promedio del nodo i para el caso de SPT, y sea N_i^D el valor de retardo promedio del nodo i para el caso de *árbol doble*. Se obtuvo entonces $C_i = \frac{N_i^D}{N_i^S}$ para cada uno de los 99 nodos. A continuación, se realizó el promedio de los 99 cocientes obtenidos, el

cual se llamará $\overline{C_1}$. Finalmente, se realizó el intervalo de confianza del 90% para $\overline{C_1}$, obteniéndose que 1.00610<$\overline{C_1}$<1.00620. Lo anterior permite concluir que, con un 90% de confianza, el retardo promedio de los paquetes enviados en *árbol doble* se incrementa 0.62% en comparación con el retardo promedio de los paquetes enviados en el árbol simple construido por SPT.

5 CONCLUSIONES Y TRABAJO FUTURO

5.1 CONCLUSIONES

El protocolo de árbol doble es un protocolo de enrutamiento que encuentra dos rutas disjuntas en nodos desde cada nodo de la red hasta la raíz en una red que sigue un patrón de comunicación de muchos a uno. El protocolo de árbol doble tiene dos fases. En la primera de ellas, los nodos encuentran las rutas mediante el intercambio de mensajes de configuración. El número de mensajes de configuración intercambiados equivale al mismo número de mensajes que se intercambiarían si se construyera un árbol típico de un solo padre en la red. La segunda fase del protocolo incluye las reglas que rigen su operación. La regla más importante es la siguiente: Cuando el primer padre de un nodo falla, el nodo desvía el tráfico hacia su padre alternativo, sin necesidad de que el árbol deba ser recalculado.

Una de las principales ventajas del protocolo de árbol doble es que garantiza una red k-connected, donde $\mathrm{k} = 2$, de manera que cada nodo posee dos padres que le proporcionan dos rutas disjuntas en nodos hasta su destino, que en este caso es siempre es la raíz. Además, el protocolo de árbol doble emplea la misma cantidad de mensajes de configuración para construirse que el protocolo de árbol más sencillo que existe, lo cual reduce significativamente el overhead en comparación con H-SPREAD [13] y DACA [14]. En comparación con el árbol de la ruta más corta, en el árbol doble los nodos solo propagan un dato adicional en sus mensajes de configuración, ya que el árbol doble emplea dos distancias distintas. Una de las distancias es empleada para escoger sus dos padres, y la otra es empleada para escoger el padre que tendrá prioridad al momento de enviarse los datos de aplicación.

Otro aspecto importante es que el protocolo de árbol doble es de topología distribuida. En el árbol doble, los nodos no necesitan información global de la red, sino solo información proveniente de sus vecinos, lo cual favorece la escalabilidad. Lo anterior no puede afirmarse con respecto a H-SPREAD [13] y DACA [14], donde cada nodo almacena sus rutas hasta la raíz incluyendo la identidad de los nodos intermediarios de la ruta.

Los resultados de las simulaciones en Omnet++ muestran que el protocolo de árbol doble es mucho más resiliente que el protocolo de árbol SPT en término de dos métricas. La primera de ellas corresponde al número de nodos conectados a la red a medida que fallan nodos progresivamente, donde los nodos fallan uno a la vez. La segunda métrica corresponde a la proporción entre nodos conectados vs nodos sobrevivientes en la red, donde un nodo conectado es un nodo que continúa teniendo una ruta hasta la raíz de acuerdo con el árbol doble construido. Por otra parte, un nodo sobreviviente corresponde a un nodo que no ha presentado fallas. La principal desventaja del protocolo de árbol doble con relación a SPT, es que en el árbol doble las rutas de los nodos tienen a ser más largas, lo cual incrementa el retardo en mensajes enviados desde los nodos hasta la raíz. Sin embargo, los resultados obtenidos en las simulaciones mostraron que ese incremento no es significativo, ya que con un 90% de confianza, no superó el 0.62%.

5.2 TRABAJO FUTURO

Como trabajo futuro, se propone generalizar el concepto de árbol doble. En el árbol doble, un requerimiento importante es que los nodos no extienden el árbol hacia la periferia de la red si no tienen dos padres que tengan distancia finita hasta la raíz. Los únicos nodos que no siguen esta regla son: La raíz y los vecinos de la raíz. Sin embargo, este requerimiento puede generalizarse de la siguiente manera: Un nodo solo puede extender el árbol si y solo si tiene M padres que le provean M rutas disjuntas en nodos hasta la raíz. Nuevamente, los únicos nodos que podrían

ser excepción a esta regla serían: la raíz y sus vecinos. Si el requerimiento mencionado se modifica como se ha explicado, podría hablarse de árboles M y de redes que sean k-connected con k=M, de manera que garanticen que cada nodo tenga M rutas disjuntas en nodos hasta la raíz. La ventaja principal de esta forma de hacer crecer el árbol sería que se emplearían la misma cantidad de mensajes de configuración que se utilizan para construir un árbol sencillo de un solo padre. Además, la topología seguiría siendo distribuida, ya que los nodos solo necesitarían información local proveniente de sus vecinos para construir el árbol, no información global.

La principal desventaja del árbol M sería que, en algunas redes poco densas, el árbol no podría crecer, ya que no podría garantizarse el requerimiento de que cada nodo tenga M padres para seguir extendiendo el árbol. Con relación a este punto, el árbol simple es ventajoso, ya que cada nodo necesita tener solo un padre para seguir extendiendo el árbol. Por tal motivo, se propone como trabajo futuro averiguar cuál es la densidad de nodos que debe haber en un espacio a fin de garantizar que el árbol M tenga una gran probabilidad de crecer adecuadamente. También se propone generalizar las reglas del protocolo de árbol doble con relación a la forma en que se escogería los M padres y la forma de escoger los padres a los cuales se envían los datos de aplicación a medida que van fallando nodos en una WSN. Finalmente, también se propone implementar el protocolo de enrutamiento de árbol M para compararlo con el protocolo N-to-1 en términos de la escalabilidad y el número de rutas disjuntas en nodos encontradas desde cada nodo de la red hasta la raíz.

BIBLIOGRAFÍA

[1] V. Potdar, A. Sharif and E. Chang, "Wireless Sensor Networks: A Survey," *2009 International Conference on Advanced Information Networking and Applications Workshops*, Bradford, 2009, pp. 636-641.

[2] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, Wireless sensor networks: a survey, Computer Networks, Volume 38, Issue 4, 15 March 2002, Pages 393-422, ISSN 1389-1286.

[3] Đurišić, M. P., Tafa, Z., Dimić, G., & Milutinović, V. (2012, June). A survey of military applications of wireless sensor networks. In *Embedded Computing (MECO), 2012 Mediterranean Conference on* (pp. 196-199). IEEE.

[4] D. Puccinelli and M. Haenggi, "Wireless sensor networks: applications and challenges of ubiquitous sensing," in *IEEE Circuits and Systems Magazine*, vol. 5, no. 3, pp. 19-31, 2005.

[5] Paradis, L., Han, Q. (2007). A survey of fault management in wireless sensor networks. Journal of Network and systems management, 15(2), 171-190.

[6] Huang, Y., Martínez, J. F., Sendra, J., López, L. (2015). Resilient Wireless Sensor Networks Using Topology Control: A Review. Sensors, 15(10), 24735-24770.

[7] J. Niu, L. Cheng, Y. Gu, L. Shu and S. K. Das, "R3E: Reliable Reactive Routing Enhancement for Wireless Sensor Networks," in *IEEE Transactions on Industrial Informatics*, vol. 10, no. 1, pp. 784-794, Feb. 2014.

[8] Al-Karaki, J. N., & Kamal, A. E. (2004). Routing techniques in wireless sensor networks: a survey. *IEEE wireless communications*, *11*(6), 6-28.

[9] Goyal, D., & Tripathy, M. R. (2012, January). Routing protocols in wireless sensor networks: a survey. In *2012 Second International Conference on Advanced Computing & Communication Technologies* (pp. 474-480). IEEE.

[10] J. Kumari and Prachi, "A comprehensive survey of routing protocols in wireless sensor networks," *2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom)*, New Delhi, 2015, pp. 325-330.

[11] Sha, K., Gehlot, J., & Greve, R. (2013). Multipath routing techniques in wireless sensor networks: A survey. *Wireless personal communications*, *70*(2), 807-829.

[12] Huang, C. K., Chang, G. Y., & Sheu, J. P. (2012, September). Load-Balanced Trees for Data Collection in Wireless Sensor Networks. In *2012 41st International Conference on Parallel Processing Workshops* (pp. 474-479). IEEE.

[13] Wenjing Lou and Younggoo Kwon, "H-SPREAD: a hybrid multipath scheme for secure and reliable data collection in wireless sensor networks," in *IEEE Transactions on Vehicular Technology*, vol. 55, no. 4, pp. 1320-1330, July 2006.

[14] S. Diaz and D. Mendez, "DACA - Disjoint Path and Clustering Algorithm for self-healing WSN," *IEEE Colombian Conference on Communication and Computing (IEEE COLCOM 2015)*, Popayan, 2015, pp. 1-5.

[15] P. Thulasiraman, S. Ramasubramanian and M. Krunz, "Disjoint Multipath Routing to Two Distinct Drains in a Multi-Drain Sensor Network," *IEEE INFOCOM 2007 - 26th IEEE International Conference on Computer Communications*, Anchorage, AK, 2007, pp. 643-651.

[16] O. Durmaz Incel, A. Ghosh, B. Krishnamachari and K. Chintalapudi, "Fast Data Collection in Tree-Based Wireless Sensor Networks," in *IEEE Transactions on Mobile Computing*, vol. 11, no. 1, pp. 86-99, Jan. 2012.

[17] N. Thepvilojanapong, Y. Tobe and K. Sezaki, "On the construction of efficient data gathering tree in wireless sensor networks," *2005 IEEE International Symposium on Circuits and Systems*, 2005, pp. 648-651 Vol. 1.

[18] Z. Han, J. Wu, J. Zhang, L. Liu and K. Tian, "A General Self-Organized Tree-Based Energy-Balance Routing Protocol for Wireless Sensor Network," in *IEEE Transactions on Nuclear Science*, vol. 61, no. 2, pp. 732-740, April 2014.

[19] Wu, Y., Mao, Z., Fahmy, S., & Shroff, N. B. (2010). Constructing maximum-lifetime data gathering forests in sensor networks. IEEE/ACM Transactions on Networking (TON), 18(5), 1571-1584.

[20] A. Saranya, R. Senthilkumaran and G. Nagarajan, "Enhancing network lifetime using tree based routing protocol in wireless sensor networks," *2015 2nd International Conference on Electronics and Communication Systems (ICECS)*, Coimbatore, 2015, pp. 1392-1396.

[21] Long, D., & Yang, Y. (2014). Low-latency SINR-based data gathering in wireless sensor networks. *IEEE Transactions on Wireless Communications*, *13*(6), 3207-3221.

[22] F. Wang and J. Liu, "Networked Wireless Sensor Data Collection: Issues, Challenges, and Approaches," in IEEE Communications Surveys & Tutorials, vol. 13, no. 4, pp. 673-687, Fourth Quarter 2011.

[23] T. W. Kuo, K. C. J. Lin and M. J. Tsai," On the Construction of Data Aggregation Tree with Minimum Energy Cost in Wireless Sensor Networks: NP-Completeness and Approximation Algorithms," in IEEE Transactions on Computers, vol. 65, no. 10, pp. 3109-3121, Oct. 1, 2016.

[24] D. Luo, X. Zhu, X. Wu and G. Chen, "Maximizing lifetime for the shortest path aggregation tree in wireless sensor networks," *2011 Proceedings IEEE INFOCOM*, Shanghai, 2011, pp. 1566-1574.

[25] J. Badarinath, S. Radhakrishnan, V. Sarangan and V. Mahendran, "Distributed Sink Tree Construction in Wireless Sensor Networks with Promiscuous Learning," *2014 IEEE 80th Vehicular Technology Conference (VTC2014-Fall)*, Vancouver, BC, 2014, pp. 1-5.

[26] Fonseca, R., Gnawali, O., Jamieson, K., Kim, S., Levis, P., & Woo, A. (2006). The collection tree protocol (CTP). *TinyOS TEP*, *123*(2).

[27] Colesanti, U., & Santini, S. (2011). The collection tree protocol for the Castalia wireless sensor networks simulator. *Department of Computer Science, ETH Zurich, Tech. Rep*, *729*.

[28] Nath, R. (2013, April). A TOSSIM based implementation and analysis of collection tree protocol in wireless sensor networks. In *Communications and Signal Processing (ICCSP), 2013 International Conference on* (pp. 484-488). IEEE.

[29] M. Moh, M. Dumont and Teng-Sheng Moh, "Evaluation of dynamic tree-based data gathering algorithms for wireless sensor networks," *Proceedings of the Fifth IEEE International Symposium on Signal Processing and Information Technology, 2005.*, Athens, 2005, pp. 170-175.

[30] Sterbenz, J.P.G.; Hutchison, D.; Çetinkaya, E.K.; Jabbar, A.; Rohrer, J.P.; Schöller, M.; Smith, P. Resilience and survivability in communication networks: Strategies, principles, and survey of disciplines. Comput. Netw. 2010, 54, 1245–1265.

[31] Mohamed Younis, Izzet F. Senturk, Kemal Akkaya, Sookyoung Lee, Fatih Senel, Topology management techniques for tolerating node failures in wireless sensor networks: A survey, Computer Networks, Volume 58, 15 January 2014, Pages 254-283, ISSN 1389-1286.

[32] Gnawali, O., Fonseca, R., Jamieson, K., Kazandjieva, M., Moss, D., & Levis, P. (2013). CTP: An efficient, robust, and reliable collection tree protocol for wireless sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, *10*(1), 16.

[33] Radi, M., Dezfouli, B., Bakar, K. A., & Lee, M. (2012). Multipath routing in wireless sensor networks: survey and research challenges. *Sensors*, *12*(1), 650-685.

[34] Kwon, S., Shin, J., Yang, D., & Kim, C. (2008, December). Practical approach to sensor data gathering. In *Intelligent Sensors, Sensor Networks and Information Processing, 2008. ISSNIP 2008. International Conference on* (pp. 545-550). IEEE.

[35] P. Djukic and S. Valaee, "Reliable packet transmissions in multipath routed wireless networks," in *IEEE Transactions on Mobile Computing*, vol. 5, no. 5, pp. 548-559, May 2006.

[36] Moonseong Kim, Euihoon Jeong, Young-Cheol Bang, Soyoung Hwang and Bongsoo Kim, "Multipath Energy-Aware Routing Protocol in Wireless Sensor Networks," *2008 5th International Conference on Networked Sensing Systems*, Kanazawa, 2008, pp. 127-130.

[37] M. Yuvaraju and K. S. S. Rani, "Secure energy efficient load balancing multipath routing protocol with power management for wireless sensor networks," *2014 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT)*, Kanyakumari, 2014, pp. 331-335.

[38] C. W. Hsu, C. S. Shieh and W. K. Lai, "A Multi-Path Routing Protocol with Reduced Control Messages for Wireless Sensor Networks," *Third International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP 2007)*, Kaohsiung, 2007, pp. 671-675.

[39] Radi, M., Dezfouli, B., Bakar, K. A., Razak, S. A., & Hwee-Pink, T. (2014). IM2PR: interference-minimized multipath routing protocol for wireless sensor networks. *Wireless Networks*, *20*(7), 1807-1823.

[40] I. Nikseresht, H. Yousefi, A. Movaghar and M. Khansari, "Interference-Aware Multipath Routing for Video Delivery in Wireless Multimedia Sensor Networks," *2012 32nd International Conference on Distributed Computing Systems Workshops*, Macau, China, 2012, pp. 216-221.

[41] J. W. Tsai and T. Moors, "Interference-aware Multipath Selection for Reliable Routing in Wireless Mesh Networks," *2007 IEEE International Conference on Mobile Adhoc and Sensor Systems*, Pisa, 2007, pp. 1-6.

[42] Sha, K., Gehlot, J., & Greve, R. (2013). Multipath routing techniques in wireless sensor networks: A survey. *Wireless personal communications*, *70*(2), 807-829.

[43] D. De Couto, D. Aguayo, J. Bicket, and R. Morris, “A high throughput path metric for multi-hop wireless routing,” in Proc. MobiCom, 2003.

[44] J. Niu, L. Cheng, Y. Gu, L. Shu and S. K. Das, "R3E: Reliable Reactive Routing Enhancement for Wireless Sensor Networks," in *IEEE Transactions on Industrial Informatics*, vol. 10, no. 1, pp. 784-794, Feb. 2014.

[45] E. Golubnichaya, "Analysis of wireless sensor networks characteristics," *2017 4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T)*, Kharkov, Ukraine, 2017, pp. 261-264.

[46] M. A. Al Rantisi, G. Mapp and O. Gemikonakli, "Simulation of an event-driven Wireless Sensor Network protocol for environmental monitoring," *Third International Conference on Future Generation Communication Technologies (FGCT 2014)*, Luton, 2014, pp. 33-38.

[47] Ding, M., & Cheng, X. Z. (2003). Aggregation tree construction in sensor networks. In *IEEE 58th vehicular technology conference (VTC)* (pp. 2168–2172). Florida, Orlando.

[48] Wu, Y., Fahmy, S., & Ness, B. S. (2008). *On the construction of a maximum lifetime data gathering tree in sensor networks: NP-completeness and approximation algorithm* (pp. 356–365). Phoenix, USA: IEEE InfoCom.

[49] Singhal, D., Barjatiya, & S., Ramamurthy, G. (2011). A novel network architecture for cognitive wireless sensor network. In *Proceedings of 2011 international conference on signal processing, communications, computing and networking technologies* (pp. 76–80) Thuckalay, India.

[50] Andreou, P., Pamboris, A., Zeinalipour, D., Chrysanthis, P.K., & Samaras, G. (2009). ETC: Energy driven tree construction in wireless sensor networks. In *IEEE 10th international conference on mobile data management: Systems, services and middleware (MDM)*, pp. 513–518.

[51] Gallagher, R.G., Humblet, P.A., Spira, M.: A Distributed Algorithm for Minimum Weight Spanning Trees. ACM Trans. on Prog. Lang. and Sys. 5 (1983)